Illumination for mixed reality of complex-to-model scenes

Katrien Jacobs

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
of the
University of London.

Department of Computer Science
University College London

October 2006

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Declaration

This document is submitted for the degree as Doctor of Philosophy in Computer Science at the Department of Computer Science at University College London. I, Katrien Jacobs, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis. The report may be freely copied and distributed provided the source is explicitly acknowledged. Data and images shown in this thesis cannot be distributed or copied separately, nor can they be manipulated and altered without written permission from the original author.

Acknowledgements

Numerous people have contributed their time, ideas, support and energy to helping me pursuing this PhD research degree. I wish it was possible to thank everyone. First and foremost, I would like to express my gratitude to Dr. Céline Loscos, my supervisor, who kept me motivated, provided valuable insights, and general advice. I would like to thank Professor Bernard Buxton, my second supervisor, for motivating me to pursue this degree in the first place, and helping me shape this thesis.

Throughout the course of this thesis, I have been supported by my colleagues at UCL and fellow researchers at various institutions. In particular I would like to thank Cameron Angus for helping me out with implementation issues involving the shadow generation application. Thanks to Danny Nahmias for providing me with the animations used in the shadow demo. I am grateful to Anders Nielsen and Jeppe Vesterbæk who have tackled several rendering problems for me and provided excellent help with the development of REFLECT. My work on HDR imaging has been supported by Greg Ward, and I thank him for our valuable discussions. I enjoyed my collaboration with Gustavo Patow and Xavier Pueyo very much, and I would like to express my gratitude for sharing their expertise and time to create our joint tutorial at Eurographics 2006.

Many proofreaders have made the writing up of this thesis and the papers during the last three years much easier. In particular I would like to thank Erik De Witte, Anthony Steed, Mel Slater, Hila Ritter Widerfield, Bernhard Spanlang, Simon Gibson, Richard Milton, Chris Christou, Cameron Angus, Anders Nielsen, Jeppe Vesterbæk, Greg Ward, Jan Kautz, Simon Prince, Simon Julier, Ashweeni Beeharee, Angus Antley, Doron Friedman, Maria Roussou, Russell Freeman, Jesper Mortensen, David Swapp, Céline Loscos and Bernard Buxton for reading my papers and thesis and giving me helpful advice.

Working in a foreign country is not always easy. Numerous visits from my family and Belgian friends were invaluable to me, as they kept me up to date with my life in Belgium. Special thanks go to Dries, as he is by far the winner of our little visiting contest.

This thesis has been financed through the Graduate School Research Award, and my gratitude goes therefore to UCL's Graduate School for allowing me to pursue this degree independently and without financial worries.

I would also like to express special gratitude to my parents and brother, for motivating and supporting me throughout my life. They have always given me the freedom to pursue my dreams, and made a home for me to go to whenever I needed it. Without their support I would not have been able to enjoy this London experience.

Finally, I would like to thank Erik De Witte, who has supported me throughout this degree, listened to my worries and dealt with my moments of stress. Completing this thesis without him would have been very difficult and therefore I dedicate this thesis to Erik.

Abstract

Illumination for mixed reality consists of simulating the illumination of 3D scenes composed with real and virtual 3D objects. Current research into illumination methods for mixed reality focuses on improving the quality of the simulated illumination, while assuming that the required input data, the scene geometry and radiance, can be accurately acquired. This thesis provides methods that reduce the importance of this condition.

As a first contribution a classification of the illumination methods for mixed reality is presented, from which can be derived that the existing methods are not robust against inaccurate input data. This limits the applicability of the previous methods to easy-to-model scenes. In complex-to-model environments, the capture of the geometry and radiance can be compromised due to camera movement, object movement, and illumination changes. Examples of such uncontrollable environments are scenes with natural lighting and deformable objects.

The influence of an incorrect geometric model and radiance distribution on the quality of the illumination methods is analysed. The problems are examined and new solutions are proposed to enable illumination simulation for mixed reality of complex-to-model scenes. Rather than improving the geometry and radiance capture using expensive equipment, this thesis provides low-cost solutions. Two solutions are separately proposed to compensate for the inaccurate geometry and for the radiance capture, while a third solution embeds the consideration of inaccurate geometry and the instability of radiance information into one single method.

The combination of the presented methods allows the simulation of consistent illumination for mixed reality without the need of increasing the efforts to capture more accurately the geometry and radiance. Making illumination methods for mixed reality more accessible and applicable to real-world environments is the key contribution of this work to the state of the art in illumination simulation for mixed reality.

Contents

Declaration 1

Abstract 3

Table of Contents 4

List of Figures 8

List of Tables 12

1 Introduction 13
- 1.1 Mixed reality . . . 13
- 1.2 Definition of illumination methods for mixed reality . . . 16
- 1.3 Problems associated with complex-to-model scenes . . . 17
- 1.4 The contributions of this thesis . . . 19
- 1.5 Thesis outline . . . 24

2 Illumination: a theoretical framework 25
- 2.1 Introduction . . . 25
- 2.2 Illumination: the radiance equation . . . 25
  - 2.2.1 Radiance and irradiance: terminology . . . 25
  - 2.2.2 Bi-directional reflectance distribution function . . . 28
  - 2.2.3 The radiance equation . . . 30
- 2.3 Illumination methods for mixed reality . . . 31
- 2.4 Inverse illumination . . . 34
  - 2.4.1 Estimating the BRDF of a material . . . 35
  - 2.4.2 Estimating the diffuse material properties . . . 36
- 2.5 High dynamic range imaging . . . 37
  - 2.5.1 High dynamic range imaging: motivation and definition . . . 38
  - 2.5.2 High dynamic range imaging: generation . . . 39
  - 2.5.3 High dynamic range imaging: limitations . . . 46
  - 2.5.4 High dynamic range display . . . 47
- 2.6 Chapter summary . . . 47

3 Review of illumination methods for mixed reality 49
- 3.1 Introduction . . . 49
- 3.2 Mixed reality . . . 50
  - 3.2.1 Mixed reality: a definition . . . 50
  - 3.2.2 Illumination methods for mixed reality . . . 52

3.2.3 Assessing the illumination methods . . . 54
3.3 Classification of illumination methods for mixed reality . . . 56
3.3.1 Choice of Classification . . . 56
3.3.2 Model of the real scene unknown, one image known . . . 59
3.3.3 Model of the real scene known, one image known . . . 61
3.3.4 Model of the real scene known, few images known . . . 68
3.3.5 Model of the real scene known, many images known . . . 71
3.4 Discussion of illumination methods . . . 73
3.4.1 Pre-processing . . . 73
3.4.2 Level of interactivity . . . 73
3.4.3 Quality obtained . . . 74
3.4.4 Overview . . . 75
3.5 This thesis within the classification . . . 75
3.6 Chapter summary . . . 77

4 Common illumination with low-detailed geometry 78
4.1 Introduction . . . 78
4.2 Problems with an inaccurate geometric reconstruction . . . 79
4.2.1 Problem description . . . 79
4.2.2 Shadows for common illumination . . . 81
4.2.3 Reconstruction inaccuracies . . . 83
4.3 Methodology . . . 86
4.4 Generating consistent shadows . . . 88
4.4.1 Shadow detection . . . 88
4.4.2 Shadow protection . . . 90
4.4.3 Shadow generation . . . 92
4.5 Results . . . 93
4.5.1 Computer augmented reality . . . 93
4.5.2 Augmented reality . . . 96
4.6 Limitations and possible improvements . . . 102
4.7 Chapter summary . . . 103

5 HDRI generation for dynamic scenes 105
5.1 Introduction . . . 105
5.2 Dynamic capture . . . 106
5.3 Practical considerations in creating HDRIs . . . 108
5.4 Methodology . . . 109
5.5 Camera movement removal . . . 110
5.6 Object movement removal . . . 112
5.6.1 Movement Removal . . . 113
5.6.2 Types of movement . . . 114
5.6.3 High contrast movement removal using a variance measure . . . 114
5.6.4 Contrast independent movement removal using entropy . . . 117
5.7 Results . . . 126
5.7.1 Camera alignment . . . 127
5.7.2 Movement removal using the variance detector . . . 132
5.7.3 Movement removal using the uncertainty detector . . . 135

5.8 Limitations and possible improvements . . . 139
5.9 Chapter summary . . . 145

6 Radiance capture of dynamically lit scenes 146
6.1 Introduction . . . 146
6.2 Dynamic illumination: a source for problems . . . 147
6.2.1 Illumination changes do happen . . . 147
6.2.2 Extracting scene radiance from one HDRI . . . 148
6.2.3 This chapter within its research field . . . 156
6.3 Methodology . . . 157
6.4 Scene capture: set-up . . . 158
6.5 Lightprobe Calibration . . . 160
6.5.1 Position calibration . . . 162
6.5.2 Sphere reflectance calibration . . . 167
6.6 Radiance registration . . . 168
6.6.1 Back-projection using OpenGL . . . 168
6.6.2 Removing pinching and distortion . . . 171
6.6.3 Numerical analysis of distortion and pinching . . . 174
6.7 Inverse Illumination . . . 175
6.7.1 Reflectance estimation per triangle . . . 176
6.7.2 Per pixel reflectance estimation . . . 178
6.7.3 Updating the BRDF using a measure for error . . . 178
6.7.4 Coherency to remove geometrical errors . . . 179
6.7.5 Rendering using PBRT . . . 180
6.8 Results for relighting applications . . . 180
6.8.1 Results for a synthetic scene . . . 181
6.8.2 Results for a real scene . . . 188
6.9 Limitations and possible improvements . . . 199
6.10 Chapter summary . . . 200

7 Conclusion 202
7.1 Introduction . . . 202
7.2 Problems revisited . . . 202
7.3 Objectives and contributions revisited . . . 203
7.4 Limitations and directions for improvement . . . 207
7.5 Future work . . . 208

Bibliography 210

A Glossary 222

B High dynamic rang image formats and encoding standards 230

C How to capture a good HDRI 231

D How to capture and model a 3 dimensional HDR model 233
D.1 Capturing input images . . . 233
D.2 Capturing lightprobe images . . . 234

D.3 Modelling using reconstruction software 235
D.3.1 Creating MCs and ICs 235
D.3.2 Creating HDRI textures 235
D.3.3 Generation of the scene file 236

E Reflect: a step-by-step manual **237**
E.1 Introduction 237
E.2 Scene description using X3D 237
E.3 Scene control 240
E.4 Lightprobe calibration 246
E.5 Lightprobe radiance registration 248
E.6 Inverse Illumination 252
E.7 Relighting 256

List of Figures

1.1 Mixed reality for educational purposes: Ename's Time window. 13
1.2 Mixed reality for medical applications: visualizing a brain tumour. 14
1.3 Mixed reality for interior design: simulating different illumination settings in a room. . . 14
1.4 Illumination for mixed reality: shadows create depth information. 15
1.5 Radiance capture hampered due to moving objects, moving camera and dynamic lighting. 18
1.6 Geometric reconstruction errors. 19
1.7 Compensating for the lack of geometry for common illumination using image processing. 21
1.8 HDRI generation: camera and object movement removal. 22
1.9 Lightprobe calibration and texture back-projection. 23

2.1 Illustrating the concepts radiance and irradiance. 27
2.2 Shadows for common illumination through scaling of the texture values. 33
2.3 An overview of a typical inverse illumination procedure. 35
2.4 Luminance values for a typical indoor/outdoor scene 39
2.5 Log exposure of film emulsions extends almost 4 orders. 40
2.6 HDRI generation: an overview. 42
2.7 Light fall-off: from irradiance to radiance. 43
2.8 HDRI generation to increase the dynamic range of an image. 44
2.9 HDRI generation to increase the dynamic range of an image: tonemapping. 45

3.1 Simplified representation of the *Reality-Virtuality Continuum*[MK94][OT99]. 50
3.2 Augmented reality: see-through display and computer augmented reality. 51
3.3 Results for Sato et al. [SSI99] 53
3.4 Results for Loscos et al. [LDR00] 54
3.5 Results for Yu et al. [YDMH99] 55
3.6 Overview of the dataflow in illumination simulation for MR. 57
3.7 Results for Nakamae et al. [NHIN86] 59
3.8 Results for Agusanto et al. [ALCS03] 60
3.9 Results for Jancene et al. [JNP+95] 62
3.10 Results for Gibson et al. [GM00] 63
3.11 Results of Debevec et al. [Deb98] 64
3.12 Results for Gibson et al. [GCHH03] 65
3.13 Results for Fournier et al. [FGR93] 66
3.14 Results for Drettakis et al. [DRB97] 67
3.15 Results for Boivin et al. [BG01] 69
3.16 Results for Loscos et al. [LFD+99] 70
3.17 Results for Gibson et al. [GHH01] 71
3.18 Results for Yu et al. [YM98] 72

4.1 Shadow simulation through scaling. 80
4.2 Shadow simulation through scaling: an example from Haller et al [HDH03]. 82
4.3 Shadow generation: the appearance of overlapping shadows. 82
4.4 Textures projected onto the geometry produce phantom shapes. 84
4.5 Augmented reality: inaccuracies in registration software. 85
4.6 The influence of the light source position on the detection of the real shadows. 86
4.7 Consistent shadow generation using a 3-step methodology. 87
4.8 Canny edge detector to detect shadow contours. 89
4.9 Shadow volumes: finding the correct scaling factor. 91
4.10 Shadow mask generation: semi-soft shadow mask. 92
4.11 3D model of a desk: modelling using ImageModeler. 94
4.12 Shadow model: instead of radiance textures, the semi-soft shadow masks are projected onto the 3D geometric model. 95
4.13 Results for Computer Augmented Reality. 97
4.14 Results for Computer Augmented Reality: benchmarking. 97
4.15 Results for real-time Augmented Reality. 98
4.16 Real-time shadow generation for augmented reality using ARToolkit: scene set-up. . . . 99
4.17 Robustness against the inaccuracies in light source position estimation. 99
4.18 Results for real-time augmented reality. 100
4.19 Limitations of shadow generation. 101

5.1 HDRI generation: object movement. 107
5.2 HDRI generation for dynamic scenes: methodology. 109
5.3 Features for alignment of different exposures. 111
5.4 Movement removal from the final HDRI through substitution. 113
5.5 Creation of a Variance Image (VI). 116
5.6 HDRI generation for dynamic scenes: methodology using the variance image. 117
5.7 Creation of an Uncertainty Image (UI). 123
5.8 Entropy as a tool to detect structures. 124
5.9 Entropy as a tool to detect movement. 125
5.10 HDRI generation for dynamic scenes: methodology using the uncertainty image. 127
5.11 Binary image tree to reduce alignment computation time. 128
5.12 HDRI generation and the influence of camera movement. 129
5.13 Camera misalignments: influence on HDRI generation 130
5.14 Camera misalignments and recovery: influence on HDRI generation 131
5.15 Recovering translational and rotational misalignments improves the quality of the generated HDRI. 131
5.16 Camera alignment: reducing the misalignments between the objects in the scene. 132
5.17 Movement detection using a Variance Image VI. 133
5.18 High contrast movement removal using variance (1). 134
5.19 High contrast movement removal using variance (2). 136
5.20 High contrast movement removal using variance (3). 137
5.21 Movement removal using an uncertainty measure (1). 138
5.22 The local entropy images highlight high entropic areas. 139
5.23 Movement removal using an uncertainty measure (2). 140
5.24 Uncertainty image and variance image: performance comparison. 141

5.25 Movement removal using an uncertainty measure (3). 142

6.1 Illumination changes visible in the texture of a 3D reconstructed scene. 149
6.2 Illumination changes during outdoor image capture. 149
6.3 An image of a reflective sphere captures omni-directional information. 150
6.4 Capturing omni-directional information with a lightprobe image. 151
6.5 Transforming the format of a lightprobe image: latitude-longitude or cube map. 152
6.6 Transformation from lightprobe image to latitude-longitude image. 153
6.7 Latitude-longitude transformation: the scene lies at infinity compared to the dimensions of the local set-up. 153
6.8 Latitude-longitude transformation: the distance between the scene and the sphere, and the camera and the sphere is infinity. The sphere is infinitesimally small. 154
6.9 Sampling the scene using a lightprobe image. 155
6.10 Relighting methodology: an overview. 158
6.11 Each input image is associated with one lightprobe image. 159
6.12 Scene composition: illumination clusters and material clusters. 161
6.13 Notations used to calculate the distance of the camera from the reflective sphere. 162
6.14 Measuring the distance of the camera to the lightprobe. 164
6.15 Notations used to find the position of the lightprobe in the 3D geometric model. 165
6.16 Calibrating the reflectance of the lightprobe image. 167
6.17 Overview of the back-projection procedure 169
6.18 Creation of the Identity Texture IT_{ij} and the position map P_k. 170
6.19 Using OpenGL to map the radiance onto the geometry. 172
6.20 Overview of the back-projection procedure: including distortion removal. 172
6.21 Alhasen's Billiard Problem. 173
6.22 Removing distortion by solving Alhasen's Billiard Problem. 174
6.23 Analysing the error introduced by distortion. 175
6.24 Inverse Illumination: an overview. 176
6.25 An input image and its identity image. 179
6.26 A synthetic scene consisting of four coloured walls, a white triangular prism and an area light source. 182
6.27 Synthetic scene modelling and back-projection. 184
6.28 Inverse illumination iteration data for a synthetic scene. 185
6.29 Assessing the influence of incorrect lightprobe positions. 186
6.30 Rendered images R_i and their associated input images I_i. 187
6.31 A real scene consisting of several different material and illumination clusters. 188
6.32 Radiance registration: back-projection with and without distortion removal. 191
6.33 A rendered image R_i, its associated original input image I_i and the error image. 192
6.34 BRDF projected onto the scene geometry: the difference between a BRDF calculated per pixel or after having applied the coherency principle. 192
6.35 Artefacts near triangle borders of a patterned MC. 194
6.36 Results inverse illumination. 194
6.37 Relighting using a novel illumination pattern. 196
6.38 Relighting after inclusion of specular objects. 197
6.39 Modelling the light sources. 198
6.40 Distortion removal on the lightprobe images. 198

D.1 Lightprobes: some practical issues. 235

E.1 Reflect: scene control (1a) 240
E.2 Reflect: scene control (1b) 241
E.3 Reflect: scene control (2a) 242
E.4 Reflect: scene control (2b) 243
E.5 Reflect: scene control (3a) 244
E.6 Reflect: scene control (3b) 245
E.7 Reflect: lightprobe calibration (1a) 246
E.8 Reflect: lightprobe calibration (1b) 247
E.9 Reflect: radiance registration (1a) 248
E.10 Reflect: radiance registration (1b) 249
E.11 Reflect: radiance registration (1c) 250
E.12 Reflect: radiance registration (1d) 251
E.13 Reflect: inverse illumination (1a) 252
E.14 Reflect: inverse illumination (1b) 253
E.15 Reflect: inverse illumination (1c) 254
E.16 Reflect: inverse illumination (1d) 255
E.17 Reflect: relighting 256

List of Tables

1.1 An overview of presentations, papers and tutorials given. 24

3.1 Overview of illumination methods for mixed reality. 76

6.1 BRDF values for the material clusters in the synthetic scene (1). 181
6.2 BRDF values for the material clusters in the synthetic scene (2). 183
6.3 BRDF values for the material clusters in the synthetic scene (3). 185
6.4 Estimating the positions of the lightprobes. 189
6.5 BRDF values for different MCs. 193

B.1 An overview of three different HDRI formats. 230
B.2 An overview of some popular HDRI encoding formats. 230

Chapter 1

Introduction

1.1 Mixed reality

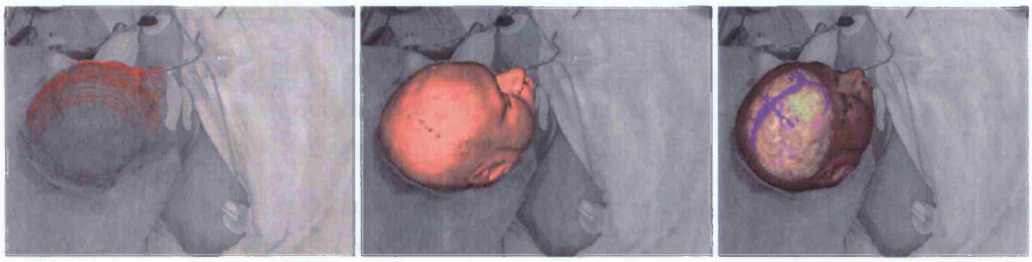
A *mixed reality (MR)* is a user-created environment in which representations for *real* and *virtual* objects are merged into a new environment [MK94]. The representation of a real object can be a photograph or video footage showing the real object, but can also simply be the projection of the object onto the retina. The representation of a virtual object can be a 3D geometric description with certain material properties or textures assigned to it. Usually an MR shows a real environment that is virtually augmented with several objects that are not originally part of this real environment. In that context, it is often referred to as an *Augmented Reality (AR)*. The augmentation can be carried out using a computer or manually. Sometimes the focus lies on creating a *realistic augmentation*, for which it is difficult to detect which object did not belong to the real scene in the first place. To create such a realistic augmentation it is important to simulate correctly the *lighting effects* between the virtual and real objects. In computer graphics, *illumination methods* are used to simulate the lighting effects in a scene. In this thesis, when we refer to an illumination method we restrict this to the class of illumination methods that can directly be applied to MR. Several illumination methods for MR exist and the applicability of these methods to real-world scenarios will be addressed and improved upon.

Various applications exist for MR in different contexts. For instance, in cultural heritage MR can be used to illustrate how a cultural site might have looked like in a different historical context [PSC01][Ena][LWR+03], as is illustrated in figure 1.1.

(a) (b) (c)

Figure 1.1: *At the Ename centre a virtual time window is used to visualize the town of Ename at 974 AC [PSC01]. (a) Visitors can look at a screen onto which a virtual model of the St. Salvator church is projected. (b) The virtual church is projected on top of live video footage of the site, the assemblage is projected onto a screen. (c) On the roof of the kiosk, a camera captures the real scene. Courtesy of Ename974[Ena]: Pam Ename and VIOE (Vlaams Instituut voor Onroerend Erfgoed).*

Another purpose of MR is to aid manufacturing processes, such as the repair of a copier machine [FMS93] or to annotate elements during the assembly of a motor [RBA+95]. Other applications use MR to highlight items previously invisible to the eye, for instance to visualize electric wiring inside a wall, to look inside cabinets, to show emergency escape routes or for computer aided surgery [Jol97], as is shown in figure 1.2.

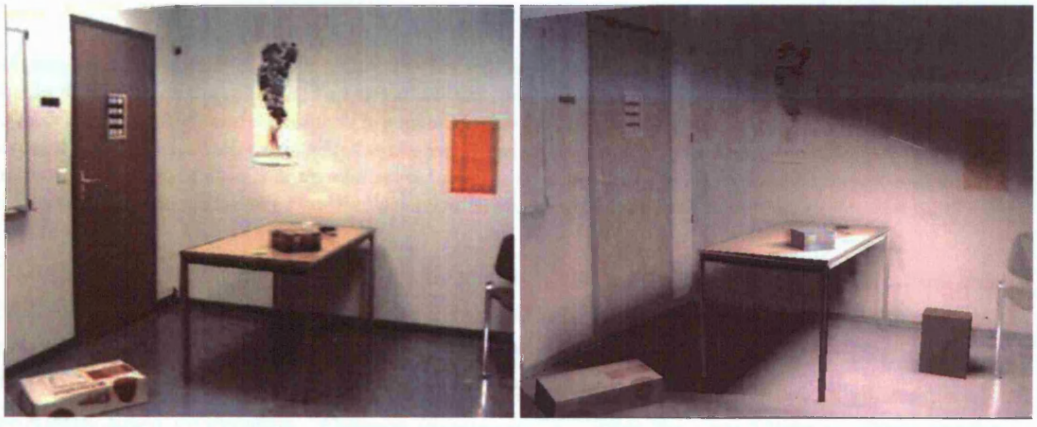


(a) (b) (c)

Figure 1.2: *Mixed reality for medical applications: visualizing a brain tumour. (a) Scanning the geometry of the patients brain. (b) Registration of the scanned and imaged surface. (c) Augmented reality display of brain data, the green mass indicates a tumour. Copyright Jolensz et al. [Jol97].*

One of the most popular and successful application fields of MR is the film and games industry. Films such as *Harry Potter*, *Lord of the rings*, *King Arthur* and many more, all make use of MR to add features, objects and/or people to certain scenes.

With a wide set of different application fields, the type of inclusion can be very different too. Some applications (e.g., surgery) require the augmentation to highlight certain areas and do not require the inclusion to be realistic, meaning that no attempt is made to visualize the virtual objects as if they were already part of the real scene in the first place. Other applications (e.g., museums, film, interior design) seek to create a new, mixed environment for which it is hard, if not impossible, to detect which object is not part of the real environment and/or whether the scene *illumination setting* has changed, see figure 1.3.



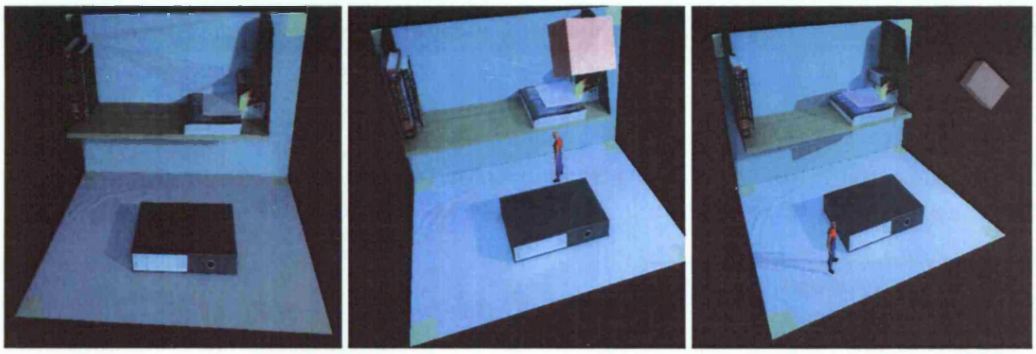
(a) (b)

Figure 1.3: *Interior design can benefit from using mixed reality to simulate different geometry and lighting designs. (a) Original image of a room containing various objects and a particular light setting. (b) Simulation of the same scene with different illumination. Interior design requires high-quality and realistic illumination simulations. Courtesy of Loscos et al. [LDR00]*

When obtaining realism is the end goal of MR, it is necessary to get several cues right:

- The virtual object needs to be correctly placed in the scene. This requires a certain level of scene calibration. Depth information about the ground plane on which the virtual object is positioned might be sufficient but information about other real scene objects is essential to recover occlusion effects.
- The virtual object needs to be registered correctly in the scene. As an example, its dimensions need to match that of the real objects.
- The illumination applied to the virtual object needs to match the illumination present in the real environment. As an example, rendering a brightly lit box in an otherwise darkly lit room will easily reveal the artificial inclusion.
- The lighting effects created by inserting a virtual object or a virtual light source need to be taken into account, and need to be *consistent* with those lighting effects already present in the real scene. As an example, the shadows cast by the virtual object onto the real scene and vice versa need to be simulated and the colour and direction of the virtual shadows needs to be according to those of the real shadows.

Figure 1.4 illustrates the importance of simulating the lighting effects, such as shadows [HH73][MR87], between virtual and real objects to provide depth information.



(a) (b) (c)

Figure 1.4: *Illumination for mixed reality. (a) The real environment consists of a desk and bookshelf with several books positioned on top of it. (b) Two virtual objects (box and avatar) are added to the real scene. The lighting effects between virtual and real objects are ignored: there are no virtual shadows visible which would have been created if the virtual objects were added to the real world as real objects. As a result it is difficult for the spectator to position the virtual objects in the real scene: there is no depth cue. (c) The shadows between virtual and real objects are generated. It is now clear that the avatar is positioned on the desk, while the box floats above the desk.*

This thesis focuses on *illumination methods* that try to maximize the level of realism of the created MR. There are several ways to match the lighting effects between real and virtual objects, and methods exist that can change the original illumination of the real objects. These illumination methods can be classified according to the type of illumination they generate: common illumination, relighting and physically-based illumination. *Common illumination* simulates the virtual lighting effects consistent with the real lighting effects already present in the scene, such as highlights and shadows. *Relighting* applies a novel illumination pattern to both real and virtual objects to make the general lighting effects consistent. *Physically-based illumination* generates common illumination or relighting using the physical laws of light and material interactions to simulate the lighting effects. These three types of illumination methods require similar input data: geometry and radiance. The *geometry* of the scene can be a general mathematical description of the position of the scene surfaces or objects. The *radiance* of

the scene is the amount of energy that leaves a surface, and can be extracted from photographs. While interesting methods exist in the literature, these methods are rarely robust against inaccuracies that exist in the geometry and radiance capture. This thesis addresses these issues related to an incorrect geometry and radiance capture and provides solutions to overcome some of the problems of the current state of the art in illumination methods.

In this chapter, the background and motivations of this thesis, its scope and the main contributions are presented. Section 1.2 explains why geometry and radiance are required to manipulate the illumination in a real environment. Then the problems that exist when applying the existing illumination methods to *complex-to-model* scenes are discussed in section 1.3. The hypothesis and contributions of this thesis are summarized in section 1.4. Finally, section 1.5 gives an overview of the structure of this thesis.

1.2 Definition of illumination methods for mixed reality

Intuitively it seems obvious that a geometric model and the scene radiance are required to simulate the effects induced by a virtual object or light source. A geometric model of the real scene is required to correctly simulate the lighting effects affected by geometric features, such as shadows between virtual and real objects and specular highlights which appear after reflecting light on a specular or glossy material. The radiance is required to visualize the illumination already present in the scene, such as real shadows, but also to analyse the material properties of the objects in the real scene.

This is theoretically formulated by the radiance or rendering equation[Kaj86]. Using the formulation from James Arvo [Arv95] this equation can be written as [Mar98]:

$$\underbrace{f}_{\substack{\text{Light reflected}\\ \text{from surfaces}}} = \underbrace{K}_{\substack{\text{How a surface}\\ \text{reflects light}}} \Big(\underbrace{h}_{\substack{\text{Direct illumination}\\ \text{from light sources}}} + \underbrace{\underbrace{G}_{\substack{\text{How the light}\\ \text{travels among the}\\ \text{surfaces}}} \underbrace{f}_{\substack{\text{Light reflected}\\ \text{from the surfaces}}}}_{\substack{\text{Indirect illumination from the}\\ \text{surfaces}}} \Big)$$

- The function f defines the light leaving a surface in any direction. Knowing f means that the scene radiance is known for every scene point and for every different angle.
- The operator K defines the reflectance of a surface, describing how it varies with the direction of the illumination and reflection angle.
- The operator G describes how the light travels between the surfaces in the scene, therefore it is dependent on the surface geometry.
- The function h describes the incident light at each scene point due to the direct lighting.

Depending on how this function is used, different problems are solved. If the radiance function f is calculated from the geometry and the reflectance, the problem is called *rendering*. If the reflectance operator K is calculated from the geometry and illumination in the scene, the problem is called *inverse illumination* [Mar98][PP03].

The illumination methods considered in this thesis find the new lighting function $\overline{f}$ resulting from the inclusion of a new (virtual) object or from changing the light sources in the scene. The inclusion of a virtual object results in a perturbation on the operator G; modifying the light sources results in a new function $\overline{h}$.

The different types of illumination methods that exist in the literature differ in how they solve the problem of calculating $\overline{f}$. Physically-based illumination uses inverse illumination to calculate the reflectance properties of the materials in the scene, which requires knowledge about f, G and h. Relighting and common illumination calculate the lighting function $\overline{f}$ based on the previous lighting function f and geometry operator G. Relighting assumes a new (virtual) function $\overline{h}$ is applied to the scene, common illumination does not change the function h. The illumination methods can further differ in the amount of knowledge they require about G and f, but in general some level of geometry and scene radiance are required.

1.3 Problems associated with complex-to-model scenes

The radiance of a scene is usually extracted from images which are projected onto the geometry of the scene. To capture the entire dynamic range of the scene such an image is usually captured as a *High Dynamic Range Image (HDRI)*. HDR imaging is a fairly new discipline which exists due to the limitations of photographic hardware. When a camera captures an image using a certain exposure setting (such as the exposure speed or aperture width) the image can show saturation or under-exposure. *Saturation* occurs when the camera's sensor saturates: it measures its physical maximum. *Under-exposure* occurs when the camera's sensor does not measure a value higher than the measuring noise. An HDRI is a floating point image for which its pixel values are proportional to the scene radiance. Often, an HDRI is generated as a weighted average of a sequence of images captured with a conventional camera using different exposures settings. The weighting removes the saturation and under-exposure that may occur in one such image. The current state of the art in HDR imaging limits the capture of HDRIs to fairly restricted environments: the camera used to capture the HDRI needs to be fixed, no objects are allowed to move in the scene and the illumination needs to be static throughout the capture. These restrictions are fairly easy to obey in controllable environments, such as an indoor scene, but are difficult to comply with in uncontrollable environments, such as an outdoor scene. An example of such a dynamic scene is shown in figure 1.5 (a). The outdoor scene is lit by a dynamic light source, and contains several moving objects such as the people walking through the scene and the water fountain. None of these dynamic objects is easy, if not impossible, to control.

Inverse illumination solves the operator K based on the geometry operator G, the direct lighting function h and the lighting function f. To perform inverse illumination, a *coherent* scene radiance needs to be available, meaning that all retrieved scene radiance has to be induced by the same global illumination, or the radiance of all scene points needs to be derived from the same function f. This implies that when the scene radiance is captured with HDRIs from different viewpoints, these images need to be captured under the same illumination settings. Scenes with dynamic illumination (e.g., in outdoor scenes) pose problems for the current state of the art in inverse illumination methods. An example of a dynamically lit outdoor scene was already shown in figure 1.5 (a). Figure 1.5 (b) shows a dynamically lit indoor scene. The sunlight that falls through the windows onto the scene gives the scene a dynamic character. When the scene radiance is extracted from a set of images captured from different viewpoints, the dynamic illumination can result in an incoherent radiance distribution.

The geometry of the real scene needs to be known in order to simulate correctly the lighting effects like shadows and highlights. The geometry can be captured using a scanner or reconstruction software. The accuracy of the geometric reconstruction needs to be high when an accurate position of a highlight or shadow needs to be retrieved. When reconstructing geometry using user-aided reconstruction software, such as ImageModeler [Rea] or PhotoModeler [Eos], the accuracy cannot always be guaranteed as it requires extensive manual input, see figure 1.6. Besides difficulties due to reconstruction software, similar

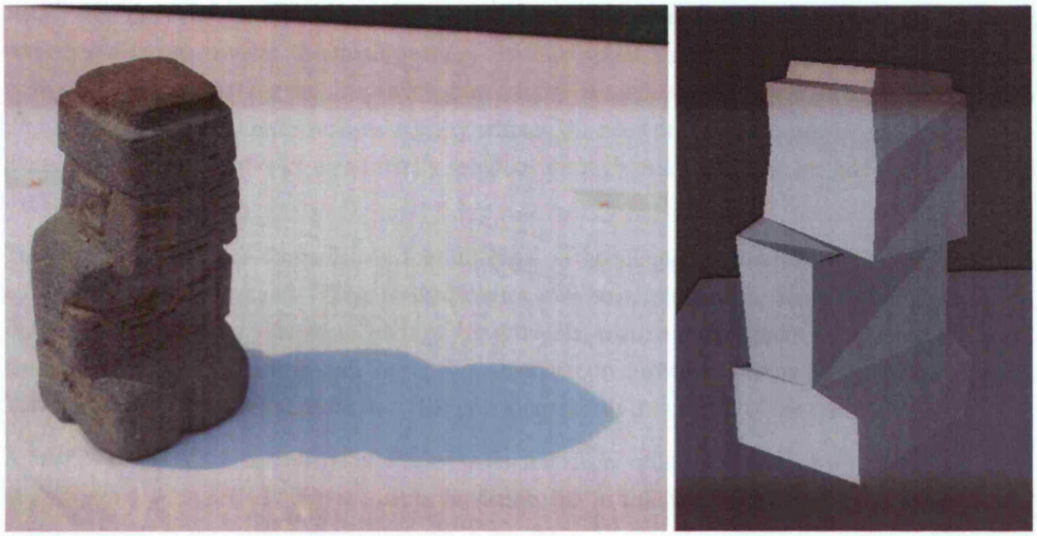
(a) (b)

Figure 1.5: *Radiance capture hampered due to moving objects, moving camera and dynamic lighting. (a) This scene contains many uncontrollable objects which can interfere with the HDRI capture, e.g., the water fountain and the people walking through the scene. (b) The light coming from the window and through the floor tiles gives the global scene illumination a strong dynamic character. While capturing the radiance for the scene with images from different viewpoints, the scene illumination may change during the different takes. The result is a global radiance inconsistency, when the different images are projected onto the geometry. This is visible when comparing the appearance of the texture within the different coloured contours.*

problems exist for other reconstruction tools, such as 3D scanners. When the capture process is lengthy, for instance, the geometry can be incorrect due to object movement or deformation. It is therefore fair to say that in uncontrollable environments, geometric reconstruction can be error-prone and unreliable.

In this thesis, emphasis is given on environments with difficult-to-capture geometry and radiance in the context of illumination generation for MR. Scenes with a difficult-to-capture geometry or radiance are called *complex-to-model*, which means that one of the following conditions applies:

- **Difficult-to-model geometry**: with user-aided reconstruction software a scene is difficult-to-model when its shape is complex (e.g., shows curvature) and needs to be derived from many different images. The modelling of a scene with a 3D scanner can be error-prone too (e.g., due to problems with sub-scattering). An incorrect model influences the inverse illumination process, the retrieval of the original lighting effects in the real scene, and the illumination generation.
- **Illumination changes**: any kind of illumination change can occur during the capture of the scene texture, typically this happens when the scene is being captured from more than one viewpoint. An illumination change is more likely to occur in an outdoor environment, due to its inherent uncontrollable illumination (e.g., the sun, clouds), but can be present in an indoor scene too. Illumination changes influence the validity of the radiance capture, which in turn influences the accuracy of the inverse illumination process and the illumination generation.
- **Object movement**: a scene can be affected by controllable (e.g., people, cars moving in the scene) and uncontrollable (e.g. movement due to the wind, birds in the sky) movement. Although indoor scenes can be affected by object movement (e.g, in public spaces), outdoor scenes are the most affected. Object movement influences the HDRI generation and therefore the validity of the scene radiance capture. Object movement or deformation also influences the feasibility of the geometric reconstruction.
- **Camera movement during the radiance capture**: using a hand-held camera to capture the radiance of a scene makes the scene modelling less restrictive. The capture will be smoother and faster,



(a) (b)

Figure 1.6: *Some objects have a complex shape and are difficult to reconstruct accurately with reconstruction software such as ImageModeler [Rea] and PhotoModeler [Eos]. (a) The statue is curved as can be verified through the curved shadow projected onto the tablecloth. (b) The 3D model consist of a triangle-based mesh, no curvatures are modelled.*

since no tripod needs to be carried around. However, capturing the images manually inevitably introduces (small) camera motion. These perturbations interfere with the HDRI generation and therefore with the validity of the scene radiance capture.

1.4 The contributions of this thesis

The previous sections discussed the importance of knowing the scene radiance and scene geometry to simulate consistent lighting effects between real and virtual objects. It was identified that capturing these input data can be difficult and error-prone in real-world scenarios. This thesis addresses these issues and improves the current state of the art in illumination methods for MR of complex-to-model scenes.

This thesis addresses the following hypothesis:

The state of the art in illumination methods for mixed reality is robust against inaccuracies in the geometry or radiance capture that exist after modelling a complex-to-model scene.

If the illumination methods are robust against such inaccuracies, this would mean that a poorly reconstructed geometry provides the same quality of MR as an accurately reconstructed geometry. The same must be true about the radiance capture. If the captured scene radiance is incoherent, this should not pose a problem for an illumination method. A first contribution of this thesis provides an overview of the illumination methods available in the literature, and from this overview it follows immediately that the illumination methods are not robust against such inaccuracies and that the hypothesis can be falsified. A theoretical analysis of the illumination methods illustrates further that an inaccurate radiance capture can and will interfere with the illumination calculations.

This led to the definition of a second hypothesis:

The robustness of the illumination methods can be improved through several useful low-cost solutions.

Rather than improving the capture of the scene radiance and scene geometry by using more expensive and reliable capture devices, this thesis uses *low-cost* equipment, such as user-aided reconstruction software and a conventional camera. Such a low-cost solution is *useful* as it improves the applicability of the illumination methods to more realistic scenery without the need of special, expensive equipment. Furthermore, this thesis provides user-friendly solutions through semi-automatic methods requiring little user-interaction.

The contributions of this thesis deliver four different methodologies to compensate for inaccurate geometry and radiance captures. These methodologies have been tested in the context of either common illumination or physically-based relighting. Nevertheless, these methodologies are general and small modifications could make them suitable for other illumination methods suffering from similar problems. This is discussed in the relevant chapters that give more details about each of the contributions.

A brief overview of the contributions is listed in the remainder of this section. A half-day tutorial given at Eurographics 2006 [LJPP06] on inverse rendering and in collaboration with University of Girona, covered several of the contributions presented in this thesis.

A Classification of illumination methods

The first contribution of this thesis is a classification of the existing methods into four groups. These four groups are defined by the amount of input data required by the illumination method:

- No geometric model known, one input image available per scene point.
- Geometric model known, one input image available per scene point.
- Geometric model known, a few input images available per scene point.
- Geometric model known, many input images available per scene point.

Several other characteristics can be assigned to each group, as they are directly related to the input data required. For instance, the quality of the resulting MR increases with the amount of input data available. Also, the more input data available, the more time-consuming the pre-processing steps are. From the classification we concluded that several problems, related to the input data, remain unsolved with the current state of the art in illumination methods, and this conclusion incited this thesis.

Chapter 3 presents the classification, and lists the main illumination methods that exists in the literature. The classification was presented as a State of the Art Report (STAR) at Eurographics 2004 [JL04], and a modified version has been published in Computer Graphics Forum [JL06].

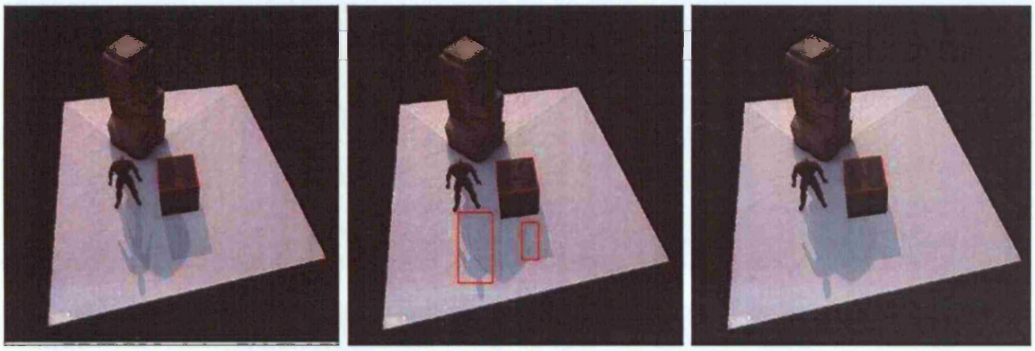
B Compensating geometry inaccuracies for common illumination

Common illumination consists of generating the lighting effects induced by the inclusion of the virtual objects into the real scene. Common illumination usually applies the lighting effects over the radiance texture retrieved during the modelling phase. For instance, virtual shadow effects are created by scaling the radiance texture of the real environment to simulate the blocking of light, without removing the real lighting effects already present in the texture.

This type of illumination simulation is sensitive to geometric inaccuracies. This can be explained as follows. Consider the case when a virtual shadow overlaps with a real shadow already visible in the scene texture. When scaling is used to simulate the virtual shadow, one has to be careful not to scale pixels that already lie inside a real shadow, as this would introduce an incorrect double scaling. To

prevent the pixels inside a real shadow from scaling, the position of the real shadow in the texture needs to be detected. Either the real shadow is detected in the radiance texture using image processing, or it is detected based on the scene geometry and the light source positions. Finding the shadows based on the geometry of the scene and the light sources is more reliable compared to image processing techniques that make no use of the geometry at all. However, when the geometric reconstruction is inaccurate, finding the shadows based on the geometry is error-prone and unreliable too.

As is discussed in chapter 4, most common illumination methods simply avoid the situation where real and virtual shadows overlap, or ignore the artefacts created. In this thesis the problems with overlapping shadows is solved by developing a novel shadow generation method that produces consistent virtual and real shadows, even when the virtual and real shadows overlap. The shadows are generated using shadow volumes to enable real-time common illumination. An example is shown in figure 1.7. The method presented in chapter 4 has been presented at Graphics Interface 2005 [JAL+05][1].



(a) (b) (c)

Figure 1.7: *(a) Generation of the virtual shadows by applying a scaling factor on the scene radiance. (b) Pixels inside the geometrical estimate of the real shadow are protected against the scaling. Due to misalignment of the real shadow and the real shadow estimate not all pixels of the real shadow are protected. (c) Using the methodology presented in chapter 4 the generated virtual shadow is consistent with the real shadow.*

C HDRI generation for dynamic scenes captured with a hand-held camera

All illumination methods require knowledge about the scene radiance. The scene radiance is usually captured using HDRIs, which effectively represent the scene radiance by means of a floating point image. An HDRI can be generated from images captured with different exposure settings or using special (expensive) HDR cameras. Both methods are sensitive to movement of the camera and the objects in the scene. This makes illumination methods, and in particular inverse illumination, unreliable when applied to uncontrollable environments such as the ones considered in this thesis. Therefore, to proof that several low-cost solutions can be developed that improve the robustness of the illumination methods against an inaccurate geometry and radiance capture, it is necessary to resolve the issues related to the HDRI generation.

Chapter 5 discusses a module that can be embedded in existing HDRI generation methods that make use of images captured with different exposure settings. This module consists of two different sub-modules. The first sub-module provides camera alignment, which enables HDRI generation from images captured with a hand-held camera. The second sub-module offers a movement removal algorithm, which enables HDRI generation from images containing moving objects. Figure 1.8 shows a preview of the results obtained. Camera movement results in misalignments between the images captured with different exposure settings, which creates incorrect ghosting reflections of the objects in the scene (a). The

[1] A demo giving an overview of the developed method can be found online at [JAL+04].

developed method aligns the sequence of images prior to weighting them into an HDRI, which effectively removes the ghosting effects (b). Moving objects create a similar effect; after the weighting certain ghosting effects can appear (c). The developed object movement removal method effectively removes such undesired effects (d).

Part of the methodology presented in chapter 5 has been presented as a Technical Sketch at Siggraph 2006 [JWL05].

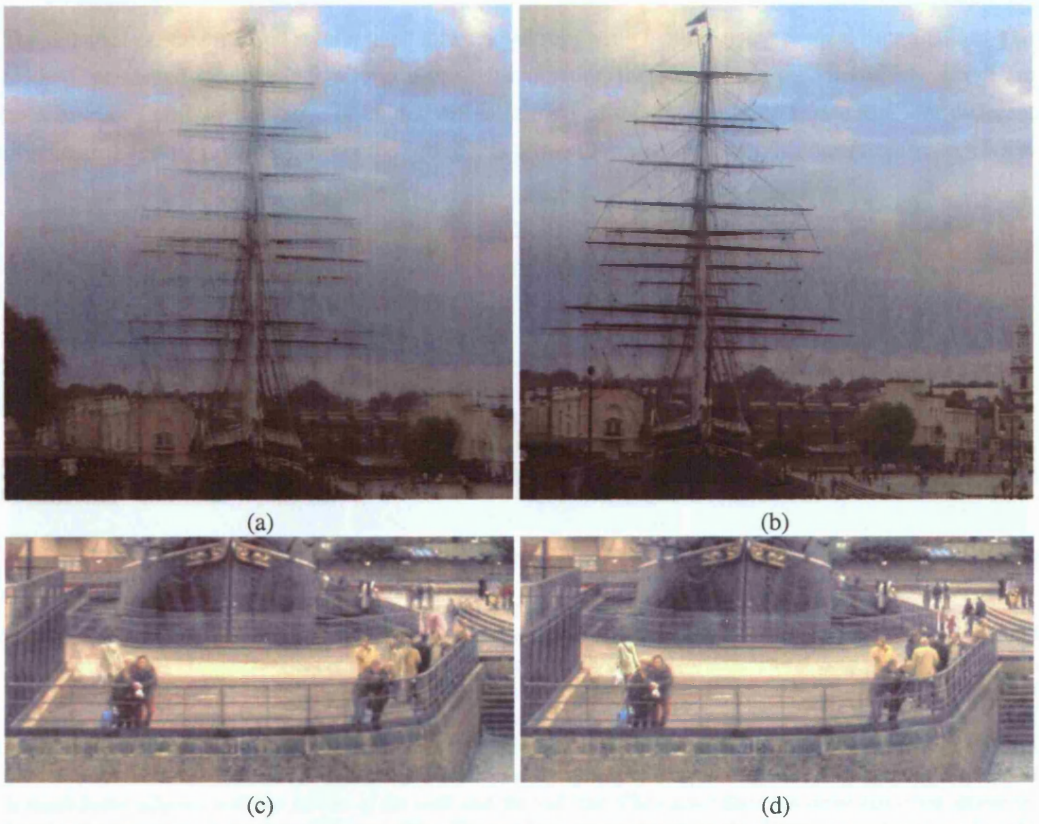


Figure 1.8: *(a) When creating an HDRI as a weighted average of images captured with a moving camera, certain ghosting effects can appear: the mast of the ship (the Cutty Sark) shows several duplications. (b) The camera alignment method developed in chapter 5 aligns the images prior to the generation of the HDRI, which effectively removes the ghosting effects: the mast of the Cutty Sark is now well-defined. (c) The movement of the people on the riverbank, creates a similar ghosting effect as in (a): copies of the woman with the white jacket and the man with the yellow jacket create duplications. (d) The developed movement removal algorithm, detects such movement and removes it from the final HDRI.*

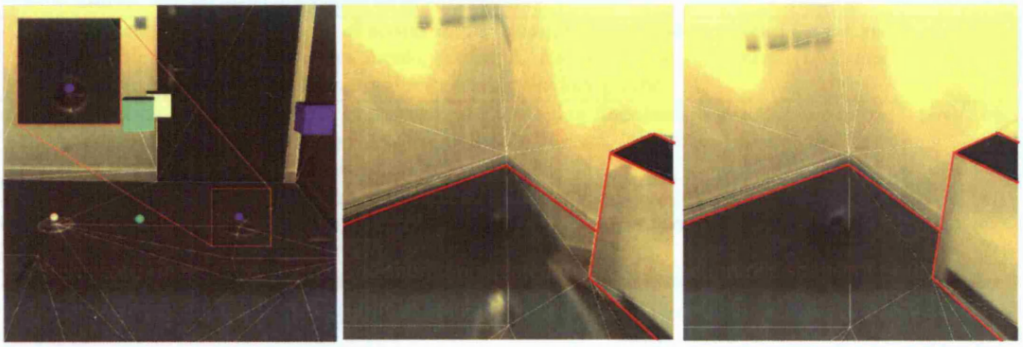
D Relighting for dynamically lit and incorrectly modelled scenes

This contribution deals with the radiance capture of scenes subject to dynamic illumination. Examples of such scenes are plentiful. Outdoor scenery is the most common example. Firstly, outdoor scenes are exposed to changing illumination throughout the day due to the dynamic position of the sun. Secondly, on cloudy days the illumination can change even more rapidly due to cloud movement. Other examples of dynamically lit scenes are indoor scenes with large windows and with uncontrollable light sources (e.g., rooms with automatic lighting systems), and scenes with large moving objects which can reflect or block the illumination in unpredictable ways.

The developed method to compensate for the illumination changes focuses on the illumination changes that occur between the capture of HDRIs from different viewpoints. More precisely the method discusses

how lightprobe images can be used and should be treated to effectively capture a snapshot of the radiance of a scene with a single HDRI. Lightprobe images are usually captured using a reflective sphere and in this thesis a method is developed to calibrate the position of the sphere based on the 3D geometry and the image data available in the lightprobe image. The projection of the radiance contained in the lightprobe image onto the scene geometry is approached mathematically. This results in a distortion-free back-projection of the radiance contained in an originally uncalibrated lightprobe image onto the real scene. An example is given in figure 1.9.

The radiance capture method is integrated into a novel relighting method using inverse illumination. The method, presented in chapter 6, allows inverse illumination for complex-to-model scenes. There are no restrictions on how the scene is lit, and different parts of the scene can be captured under different illumination settings. This improves the robustness of the illumination methods considerably.



(a) (b) (c)

Figure 1.9: *These images show a 3D scene consisting of four walls, a radiator, a door, and blue carpet. (a) The position of 3 reflective spheres (represented by coloured spheres) and the associated camera positions (represented by coloured boxes) in the 3D scene are retrieved using a novel calibration algorithm. The accuracy of the calibration can be assessed from their modelled positions visible in the projected wireframe (see inset). (b) The back-projection of a calibrated lightprobe image onto surfaces at a distance of* $\approx 50cm$ *without considering distortion effects (often introduced when ignoring the finite set-up of the scene). The distortion is mainly visible on the carpet, radiator and border of the wall. (c) The back-projection of a calibrated lightprobe after removing distortion effects. The texture is much better aligned with the border of the wall and the radiator. The carpet does not show distortion anymore, but the shadow of the lightprobe is now visible. The red lines highlight where the distortion removal can best be assessed.*

E Compensating geometry inaccuracies for inverse illumination

The inverse illumination process, discussed in the previous subsection, is not only sensitive to an inaccurate radiance capture, but also to geometrical errors. In the relighting method discussed in chapter 6, a coherency principle is presented that effectively removes the artefacts that result from geometrical errors, as well as errors due to using an incorrect illumination model. The result is an inverse illumination method that can be applied to complex-to-model scenes.

The last two contributions have been presented as one relighting application for dynamically lit scenes as a poster at Eurographics' Rendering Workshop 2006, and is registered as a Technical Report at University College London[JNVL06a][2]. Appendix E provides a manual for the software system REFLECT designed for this purpose. The resulting relighting method, in combination with the previously summarized contributions, can be applied to scenes lit by a dynamic light source, subject to object movement, and poorly reconstructed. This combination allows physically-based illumination for complex-to-model scenes such as outlined in section 1.3, using low-cost equipment.

[2] A demo giving an overview of the developed method can be found online at [JNVL06b].

F Presentations, posters, papers and tutorials

Table 1.1 lists the presentations, peer-reviewed papers published, and tutorial workshops given in the relation to this thesis.

2004	**Relighting Outdoor Scenes**, *Katrien Jacobs and Céline Loscos*, British Machine Vision Association, Vision, Video and Graphics: one day symposium	
2004	**Classification of illumination methods for mixed reality**, *Katrien Jacobs and Céline Loscos*, State of the Art Report, Eurographics 2005	[JL04]
2005	**Automatic consistent shadow generation for augmented reality**, *Katrien Jacobs, Cameron Angus, Céline Loscos, Jean-Daniel Nahmias, Alex Reche and Anthony Steed*, Conference paper, Graphics Interface 2005	[JAL+05]
2005	**Automatic high dynamic range image generation for dynamic scenes**, *Katrien Jacobs, Greg Ward and Céline Loscos*, Technical Sketch, Siggraph 2005	[JWL05]
2006	**Classification of illumination methods for mixed reality**, *Katrien Jacobs and Céline Loscos*, Journal paper, Computer Graphics Forum	[JL06]
2006	**Coherent radiance capture of scenes under changing illumination conditions for relighting applications**, *Katrien Jacobs, Anders H. Nielsen, Jeppe Vesterbaek and Céline Loscos*, Research poster, Eurographics Rendering Workshop 2006	
2006	**Coherent radiance capture of scenes under changing illumination conditions for relighting applications**, *Katrien Jacobs, Anders H. Nielsen, Jeppe Vesterbaek and Céline Loscos*, Technical Report, University College London	[JNVL06a]
2006	**Inverse Illumination: from concept to applications**, *Céline Loscos, Katrien Jacobs, Gustavo Patow and Xavier Pueyo*, Half day Tutorial, Eurographics 2006	[LJPP06]

Table 1.1: *An overview of presentations, papers and tutorials given or published in the context of this thesis.*

1.5 Thesis outline

This thesis is structured as follows. Chapter 2 provides a theoretical background into the illumination methods for mixed reality, and defines the necessary terminology used throughout this thesis. Chapter 3 gives an in-depth literature overview, providing a classification of the most relevant illumination methods for mixed reality. In chapter 4 a new common illumination method that generates consistent virtual shadows for poorly reconstructed scenes is presented. Chapter 5 improves the state of the art in HDRI generation and provides a solution for camera movement and moving objects occurring during the HDRI capture. Chapter 6 presents a relighting application which is robust against scene illumination changes and inaccuracies in the geometric reconstruction. Finally, chapter 7 gives a summary of this thesis, reviews how well its objectives have been met, and provides some directions for future work.

Additional information about certain concepts or methodologies presented in this thesis, is given in the appendices. Appendix A provides a glossary with the terminology used throughout this thesis. Appendix B provides an overview of some commonly used HDRI formats. Appendices C and D provide practical information about capturing an HDRI and an HDRI textured 3D model respectively. Finally, appendix E contains a manual that describes the input required for the developed relighting system, REFLECT, along with a user-guide for the developed software package.

Chapter 2

Illumination: a theoretical framework

2.1 Introduction

The radiance distribution in a scene is defined by the light sources present, the scene geometry and the material properties of the objects in the scene. The process of distributing the radiance can also be described as an exchange of energy, which results in a stable solution defined by the *radiance or rendering equation* [Kaj86]. This equation defines the radiance of each scene point, based on the light sources, the scene geometry and the material properties. Illumination methods use the radiance equation to predict the influence of a foreign object or light source in a given environment.

This chapter gives a theoretical overview of the radiance equation and explains its dependency on the scene geometry and radiance, and the material properties of the objects in the scene. A general framework to simulate common illumination, relighting and physically-based illumination for mixed reality is given, which should clarify the dependency of the illumination methods on the scene geometry and radiance. The scene radiance can be captured using an HDRI. In this chapter a definition of HDR imaging is given, along with an explanation of how an HDRI is generated and can be used to capture the scene radiance.

The objective of this chapter is to understand the different aspects of the illumination methods, and to understand the influence of the accuracy of the input data. The presented terminology is used throughout this thesis.

This chapter is organised as follows. Section 2.2 presents definitions for concepts such as radiance, irradiance and BRDF. Section 2.3 explains how the illumination methods for mixed reality depend on the radiance equation. Section 2.4 gives a general framework to perform inverse illumination in order to calculate the material properties of the objects in the real scene. Section 2.5 provides an overview of HDR imaging. A chapter summary is given in section 2.6.

2.2 Illumination: the radiance equation

2.2.1 Radiance and irradiance: terminology

Visible light is electromagnetic radiation with wavelengths between 400nm and 700nm. The colour of light depends on its wavelength, it is bluish near 400nm and reddish at 700nm. The distribution of light in a scene needs to obey the laws of dynamic equilibrium and conservation of energy [SSC02]:

1. Dynamic equilibrium: the flux in a volume must be in dynamic equilibrium. This means that the

overall distribution of energy in a volume must remain constant if nothing changes. Some parts of the volume cannot become darker or brighter without a specific reason.

2. Law of conservation of energy: the total amount of energy leaving a volume must be equal to the amount of incoming energy in the volume minus the amount of energy absorbed by the volume.

In computer graphics, a few more assumptions are often made: photons with different wavelengths do not interfere, light travels in a vacuum, a system is time invariant when no physical scene changes occur [SSC02]. Since light is an electro-magnetic wave it needs to obey the four Maxwell equations at all time [Max64]. With this in mind, it is now possible to define the basic concepts that are needed to describe the flow of light and the resulting equilibrium distribution of the energy in a scene.

A Points and surfaces

In principle, a scene consists of solid objects, bounded by surfaces that can be defined by a mathematical 3D equation. The surfaces of a scene consist of a continuous set of points p, for which a normal $\overrightarrow{n}$ can be defined. When in this thesis a reference is made to a scene point, in fact a 3D scene surface point is meant because, when ignoring translucent effects, in general the non-surface points are of less importance for illumination methods.

For some scene points, no normal can be defined, for instance at discontinuities (e.g., the edges of a cube). Though we could define a proper mathematical expression of a normal, this is beyond the scope of this thesis, as usually discontinuities do not pose a problem when developing general algorithms for illumination methods and abstraction can be made of such discontinuities.

B Radiant energy and flux

Radiant energy is the energy of an electro-magnetic wave. The radiant energy is denoted by Q and is measured in Joules. The radiant energy flowing through (leaving or arriving) a surface per unit time is called flux:

$$\Phi = \frac{dQ}{dt} \tag{2.1}$$

The unit of flux is $\frac{Joules}{Seconds}$, which is also referred to as Watt.

C Radiance

Radiance is the amount of flux leaving a surface per unit projected area of the surface and per unit solid angle. The unit of radiance is $\frac{Watt}{Steradian \times m^2}$.

A graphical interpretation of radiance is illustrated in figure 2.1 (a). The surface from which the flux leaves has an area dA, the normal on the surface is given by the vector $\overrightarrow{n}$, the solid angle through which the energy leaves is $d\omega$, the letter d indicates that differential values are considered, θ is the angle between the normal $\overrightarrow{n}$ and the solid angle $d\omega$. If radiance is denoted by a capital letter L, the following relation between flux and radiance can be defined:

$$d^2\Phi = L dA \cos\theta d\omega, \tag{2.2}$$

since the projected area of dA in the direction of $d\omega$ is given by $dA\cos\theta$. If we let dA and $d\omega$ become infinitesimally small, $d^2\Phi$ can be correctly interpreted as flux along a ray. The radiance in a system depends on the position in the scene, denoted by a point p, and on the outgoing direction in which the flux leaves, denoted by a differential solid angle $d\omega$, therefore a more correct expression for L would be $L(p, d\omega)$.

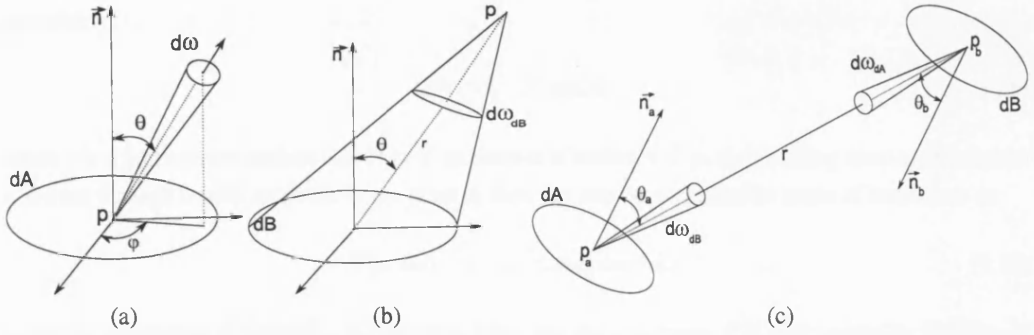


Figure 2.1: *(a) Radiance is the amount of flux* leaving *a surface dA, per unit area of the surface and per unit solid angle dω. (b) The solid angle $d\omega_{dB}$ can be defined in terms of a differential area dB. (c) Interaction between two infinitesimal patches dA and dB.*

A differential solid angle has the meaning of a direction, and it can entirely be defined by an azimuth angle ϕ and an incident angle θ, see figure 2.1 (a), where θ and ϕ are the spherical coordinates of the direction $d\omega$ in a coordinate system aligned with the normal at p. In fact $d\omega = sin\theta d\theta d\phi$. There is no need for the third spherical coordinate r, since a direction has no length. Sometimes $d\omega$ is represented as a normalized 3D vector $\omega = [\omega_x, \omega_y, \omega_z]$, where the vector defines the direction. Figure 2.1 (b) illustrates another definition of the solid angle:

$$d\omega_{dB} = \frac{dBcos\theta}{r^2} \tag{2.3}$$

In figure 2.1 (c), the energy interaction between two patches is illustrated. Suppose an amount of flux is leaving in the direction $d\omega_{dB}$, yielding an amount of radiance equal to $L(p_a, d\omega_{dB}) = L$. The amount of flux travelling between the points p_a and p_b can be written as:

$$\begin{aligned} d\Phi(p_a, p_b) &= LdA\cos\theta_A d\omega_{dB} && (2.4)\\ &= LdA\cos\theta_A \frac{dB\cos\theta_B}{r^2} && (2.5)\\ &= LdB\cos\theta_B \frac{dA\cos\theta_A}{r^2} && (2.6)\\ &= LdB\cos\theta_B d\omega_{dA} && (2.7)\\ &= d\Phi(p_b, p_a) && (2.8) \end{aligned}$$

Equation 2.5 shows that the amount of flux travelling between two patches (through an area), decreases with the square of the distance between the patches, while the radiance along the ray (through a solid angle) remains constant, as long as differential solid angles are considered. Equation 2.7 can be interpreted as if an amount of flux is leaving from the point p_b to the point p_a. From these equations it is clear that reversing the direction of the energy does not influence the amount of energy that flows. Saying that the energy flows from p_b to p_a does not change the amount of energy that flows along the ray. In general this means that if the direction of the energy flow is reversed for the entire system, that the same equations will still describe the flow of energy, albeit in the opposite direction.

D Irradiance

The irradiance at a surface is the amount of energy arriving at the surface per unit area of the surface and is represented by the capital letter E, the unit of irradiance is $\frac{Watt}{m^2}$. This translates in mathematical

terms to:

$$d\Phi = E(p)dA, \tag{2.9}$$

where p is a point on the surface dA. Now if an amount of radiance $L(p, d\omega)$, coming from another point q, arrives through a solid angle $d\omega$ at the point p, then this can be expressed in terms of irradiance as:

$$E(p, d\omega) = L(p, d\omega)\cos\theta d\omega, \tag{2.10}$$

where the definitions of figure 2.1 (a) are used. Note that the irradiance E is now written in function of p and $d\omega$ radiance. This is because it is possible that more radiance arrives at p from other directions than $d\omega$. Also note that the $L(p, d\omega)$ is described in the coordinate system positioned at p instead of the coordinate system positioned at q. This is only possible because the radiance along a ray, connecting the two points, remains constant and is not influenced by the distance between the two points. This equation illustrates the interchangeability between radiance and irradiance at a point in a certain direction, where the irradiance is *arriving* at the point and the radiance is *leaving* from *another* point.

E Intensity

The intensity (I) of a point in a scene is defined as the flux per solid angle. The unit of intensity is $\frac{Watt}{Steradian}$. Intensity is related to radiance in the following way:

$$I = L(p, d\omega)dA\cos\theta \tag{2.11}$$

F Wavelength dependency

The definitions given above are not only dependent on the position p and the direction ω, but also depend on the wavelengths of the light that hits, or leaves, a surface. The definitions of radiant energy, flux, radiance, irradiance and intensity are actually integrated quantities over the possible wavelengths. The radiant energy can be described for each wavelength, but it is more common to define it over the entire interval of wavelengths. When considering photometric values this means integrated over the interval $[400nm : 700nm]$. There may be some inter-wavelength dependency, but we make the reasonable assumption that the radiant energy can be calculated per wavelength. This assumption rules out fluorescence.

An image usually represents its pixel values as a triplet: $[R, G, B]$. Each component (R, G and B) is the result of an integration of the incoming electromagnetic wave with a certain response curve, which differs for the R, G and B channel. A camera's sensors measures irradiance, and due to the integration over the response curve, an image's pixel values actually represent a triplet of irradiance values $[E_R, E_G, E_B]$. This triplet can be interpreted as a decomposition of the original irradiance into three components.

In the equations that follow, the wavelength dependency is implicitly assumed, though not explicitly mentioned. Or, when discussing radiance or irradiance values, the underlying decomposition into three colour components is implicitly assumed.

2.2.2 Bi-directional reflectance distribution function

Now that the concepts radiance and irradiance are defined here, it is possible to give a definition of the *Bi-directional Distribution Function* or *BRDF* of a point or infinitesimal surface. The BRDF of a material at a point p defines how much radiance $L(p, \omega_r)$ at point p and in an outgoing direction ω_r

results from reflecting the amount of irradiance $E(p, \omega_i)$ at point p coming from the direction ω_i. The BRDF of a certain point is denoted as $f(p, \omega_i, \omega_r)$ and its unit is by definition $\frac{1}{Steradian}$, but has no physical meaning.

A mathematical definition of the BRDF can be written as:

$$L(p, \omega_r) = f(p, \omega_i, \omega_r) E(p, \omega_i) \tag{2.12}$$

In other words, the BRDF relates the incident energy $E(p, \omega_i)$ in a solid angle ω_i with the outgoing energy $L(p, \omega_r)$ in a solid angle ω_r at a point p with BRDF $f(p, \omega_i, \omega_r)$. Due to the law of conservation of energy, the range of the BRDF is restricted to the interval $[0, 1]$. The BRDF depends on the incident direction, the outgoing direction, the position in the scene and the wavelength of the incoming light. A BRDF can also be subject to sub-scattering, but in this thesis sub-scattering is ignored.

Various different mathematical models exist for the BRDF of a material. The complexity of the BRDF model varies between different materials. A survey of several commonly used BRDF models is given in [Sch94], but this survey might be slightly outdated as in the mean time several new models have been developed.

More recently Ngan et al. [NDM05] made an experimental analysis of the most popular BRDF models. They divide the existing models into two groups. A first group is derived from the phenomena that the BRDF introduces (such as highlights), examples are [Pho75][Bli77][Lar92][LFTG97][AS00]. The models are derived from experiments, and are designed to match the lighting effects perceived. The second group of BRDF models are derived based on the physical aspects involved, examples are [CT81] [HTSG91][HHP+92].

Without going into too much detail, it is important to explain a few popular terms, often associated with BRDF models. The BRDF of a scene point p of a certain material is a 2D function, and can be *isotropic* or *anisotropic*:

- **Isotropic:** the BRDF function at a point p is rotationally symmetric around the normal (e.g., paint). The BRDF is still dependent on the viewing angle within the plane of the 2D function.
- **Anisotropic:** the BRDF at point p is truly a 2D function and is not symmetric around the normal. In other words, when rotating around the normal, the BRDF value changes (e.g., brushed metal, wood).

A material can either be *diffuse*, *specular*, *glossy* or *transparent*:

- **Diffuse:** the material reflects the light uniformly over the entire hemisphere. Such a material's BRDF is a constant, and independent of the direction $d\omega_r$.
- **Specular:** a purely specular material reflects light from a certain direction into one outgoing direction. According to Snell's law, the outgoing direction should be equal to the incoming direction.
- **Glossy:** most surfaces are not ideal diffuse reflectors, nor ideal specular reflectors, but are usually a combination of the two. Sometimes this is referred to by the term glossy.
- **Transparent:** a transparent material can reflect light over the entire 4π space, while for other materials the BRDF is usually only defined for the hemisphere. A transparent material exhibits also diffuse, specular and glossy characteristics. In this thesis we ignore transparency effects, and limit our discussions to solid materials with a diffuse, specular and glossy character.

As an illustration two different BRDF models are discussed in this section. The first is the Phong model [Pho75], the second is the Ward model [Lar92]. The Phong model is an empirical model, based on observations, which splits the diffuse and specular components into two separate functions. The BRDF can be represented as:

$$f(p, \omega_i, \omega_r) = \frac{\rho_d}{\pi} + \rho_s \frac{(n \cdot h)^m}{n \cdot w_i}, \tag{2.13}$$

$$h = \frac{w_i + w_r}{|w_i + w_r|} \tag{2.14}$$

$$|w_i| = |w_r| = |n| = 1 \tag{2.15}$$

where ω_i, ω_r, n and h are normalized vectors describing a direction. The diffuse term $\frac{\rho_d}{\pi}$ obeys *Lambert's law*, which states that the energy leaving is proportional to the cosine of the incoming angle, for diffuse reflection. The specular term contains an exponential component, m which defines the *shininess*. The Phong model presented in these equations is isotropic, an anisotropic adaptation to this model is developed by Ashikhmin et al. [AS00].

The Ward model proposes the following mathematical model for an isotropic BRDF:

$$f_{spec}(p, \omega_i, \omega_r) = \frac{\rho_d}{\pi} + \rho_s K(\alpha, \omega_i, \omega_r), \tag{2.16}$$

$$K(\alpha, \omega_i, \omega_r) = \frac{1}{\sqrt{\cos\theta_i \cos\theta_r}} \frac{exp[-tan^2\delta/\alpha^2]}{4\pi\alpha^2} \tag{2.17}$$

with θ_i, θ_r derived from ω_i, ω_r, and δ the bisector of the angles θ_i and θ_r. ρ_d and ρ_s are respectively the diffuse and specular reflectance of the material, α is the surface roughness factor. For isotropic materials the surface roughness factor is a scalar. For anisotropic materials this factor would be a vector with two components (an X and Y component) [Lar92]. Walter et al. [Wal05] provides extra notes on the implementation issues of this model and defines its probability density function useful for Monte Carlo sampling applications.

2.2.3 The radiance equation

It has been mentioned in the introduction that the radiance equation describes the energy distribution in a scene. If the spectral energy of all points in a scene are known, the appearance (colour) of all points in the scene is known and the scene can be perfectly and accurately rendered.

The radiance equation given in chapter 1, can be redefined in terms of radiance and BRDFs. It defines that the total radiance at a point p in a certain direction ω, must equal the sum of all energy emitted by the object at p in the direction ω and the total amount of irradiance arriving at p that is reflected into the direction ω. The total amount of irradiance at the point p is equal to the radiance from all other points of the scene. In mathematical terms this translates to:

$$L(p, \omega) = L_e(p, \omega) + \int_\Omega f(p, \omega_i, \omega) L(p, \omega_i) \cos\theta_i d\omega_i, \tag{2.18}$$

where $L_e(p, \omega)$ stands for the emitted radiance at point p in direction ω, the radiance $L(p, \omega_i)$ is the radiance coming from another point in the scene in a certain direction ω_i expressed in a coordinate system positioned at p, and Ω represents the hemisphere. The integration measure $d\omega_i$ is equal to $sin(\theta_i)d\theta_i d\phi_i$, where θ_i and ϕ_i are the spherical coordinates of the azimuth and elevation respectively. Therefore we

can re-write equation 2.18 as:

$$L(p,\omega) = L_e(p,\omega) + \int_0^{2\pi}\int_0^{\frac{\pi}{2}} f(p,\omega_i,\omega)L(p,\omega_i)\cos\theta_i sin\theta_i d\theta_i d\phi_i, \quad (2.19)$$

To calculate the radiance of a point p into a certain direction ω, equation 2.19 needs to be solved. In order to do so, the scene geometry (the points p), the reflectance functions $f(\cdot)$, and the scene radiance for all other scene points, at least those visible from p, need to be known. When a light source is switched on in a scene, its energy is distributed into the scene, and the system will converge to a stable solution defined by the system of equations consisting of the radiance equation for all scene points p and in all directions ω.

Illumination methods for mixed reality try to estimate the radiance of the points in a scene, after the inclusion of a virtual object or light source. In chapter 3 it is explained that certain illumination methods require more input than others to achieve this. It is obvious that the more information is available the more precise the radiance estimate will be. Ideally all parameters are known (geometry, scene radiance and reflectance functions) and in that case a precise radiance estimate can be made. Sections 2.3 and 2.4 discuss further how illumination methods and inverse illumination operate, based on the defined radiance equation.

2.3 Illumination methods for mixed reality

Three different types of illumination methods for mixed reality, often simply referred to by illumination methods, exist. In short these are: common illumination (virtual and real objects share a consistent illumination, the illumination of the real objects remains unchanged), relighting (virtual and real objects share a consistent illumination, the illumination of the real objects can be changed) and physically-based illumination (similar to relighting or common illumination except for that it calculates the reflectance values of the real objects in the scene to simulate the lighting effects according to the physical laws of material/light/geometry interaction). Chapter 3 provides an overview of the illumination methods available in the literature. Each of these methods presented in that chapter has its unique features, nevertheless they often share the same underlying ideas, based on the radiance equation. This section provides a theoretical framework that lies behind most of these illumination methods. Rather then providing an in-depth derivation of the methods, a brief structure is provided that should clarify the complications these methods face if certain data is unavailable.

In computer graphics, a scene is often represented as a discrete set of points, therefore the integral in equation 2.18 can be replaced by a sum:

$$L(p,\omega) = L_e(p,\omega) + \sum_{p' \in scene} V_{pp'} f(p,\omega_{p'},\omega)L(p,\omega_{p'})\cos\theta_{pp'} \quad (2.20)$$

$$V_{pp'} = \begin{cases} 0 \text{ if points p and p' are invisible to each other,} \\ 1 \text{ if points p and p' are visible to each other.} \end{cases}$$

where $\omega_{p'}$ is the direction from p to p' and $\theta_{pp'}$ the azimuth angle of $\omega_{p'}$. In this equation the function $V_{pp'}$ is a visibility function. For most scene points (excluding light sources), we can assume that the emittance $L_e(p,\omega)$ is zero, while a light source emits an amount L_e and has zero reflectance. Therefore,

for general scene points, excluding points p on a light source, we can re-write equation 2.20 as:

$$L(p,\omega) = \sum_{p' \in scene} V_{pp'} f(p,\omega_{p'},\omega) L(p,\omega_{p'}) \cos\theta_{pp'} \tag{2.21}$$

$$V_{pp'} = \begin{cases} 0 \text{ if points p and p' are invisible to each other,} \\ 1 \text{ if points p and p' are visible to each other.} \end{cases}$$

If a virtual object is inserted into the scene, its presence will influence the visibility function $V_{pp'}$. Introducing a new visibility function will immediately change certain radiance values $L(p)$ which will destabilize the entire system of equations defined by the radiance equations for all scene points p and in all directions ω. Hence to simulate the new radiance distribution, a new stable solution for the radiance equations defined for all scene points needs to be calculated.

When certain (or all) light sources are switched off, and new light sources are introduced into the scene, a similar effect takes place. For some points the radiance values change, which destabilizes the current radiance distribution as defined by the radiance equations for all scene points p in all directions ω. This requires, again, a new radiance distribution to be calculated.

Simulating both effects, which can be labelled as either common illumination or relighting, therefore requires a re-calculation of the set of radiance equations defined by all scene points p in all directions ω. Some methods limit these calculations to points near the disturbance (e.g., [Deb98]). Nevertheless to re-calculate the system of equations it is important to know some level of the scene geometry, the scene radiance and the scene BRDF.

The requirement to know the BRDF values of the scene is less stringent than the requirement to know the radiance and the geometry. If just simulating common illumination, a scaling of the radiance values within a *new* virtual shadow can already induce the effect of a real shadow (e.g., [HDH03]). This can be explained using figure 2.2, which shows a 2D graphical interpretation of how light reaches a certain scene point p. The radiance of a point p in a certain direction is defined by its irradiance and reflectance function. In (a) a point p receives light from a flat light source LS (shown in grey) which can be considered as direct lighting, and from the hemisphere which reflects this direct lighting and therefore can be considered as indirect lighting. When a virtual object is inserted in the scene as shown in (b), the amount of light arriving at p is reduced. The situation in (a) and (b) can respectively be written as:

$$L(p,\omega) = \sum_{p' \in scene} V_{pp'} f(p,\omega_{p'},\omega) L(p,\omega_{p'}) \cos\theta_{pp'} \tag{2.22}$$

$$\overline{L}(p,\omega) = \sum_{p' \in scene} \overline{V_{pp'}} f(p,\omega_{p'},\omega) \overline{L}(p,\omega_{p'}) \cos\theta_{pp'} \tag{2.23}$$

with $\overline{V_{pp'}}$ the new visibility function and $\overline{L}$ the new radiance distribution resulting from the inclusion of the virtual object. If we would consider direct lighting only and ignore indirect lighting as shown in (c) and (d), the radiance equations can be written before the inclusion of the virtual object (c) as:

$$\begin{aligned} L(p,\omega) &= \underbrace{\sum_{p' \in LS} V_{pp'} f(p,p') L(p,\omega_{p'}) \cos\theta_{pp'}}_{\text{Direct lighting}} + \underbrace{\sum_{p' \in scene \setminus LS} V_{pp'} f(p,p') L(p,\omega_{p'}) \cos\theta_{pp'}}_{\text{Indirect lighting}} \\ &= L_{direct} + L_{indirect} \\ &\approx L_{direct} \end{aligned}$$

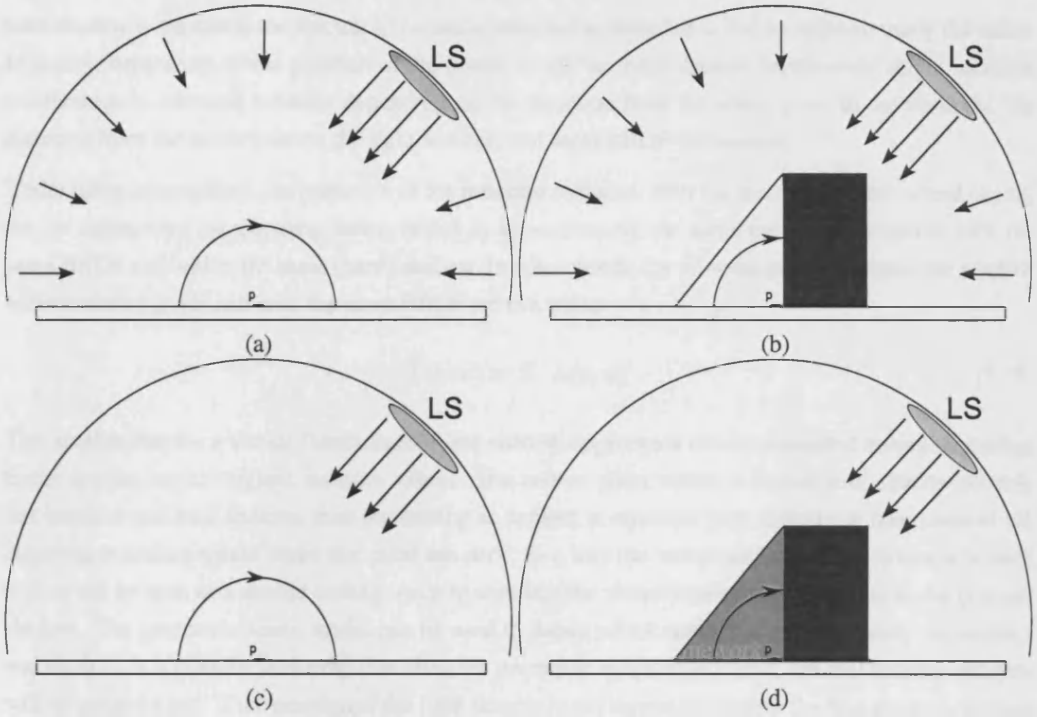


Figure 2.2: *These images show 4 different scenarios representing light arriving at scene point p. In (a) the light arrives at point p from all directions, most light arrives from the greyish flat light source (LS), which can be considered direct lighting, but some indirect lighting arrives from other parts of the scene. (b) After inclusion of a virtual object (black box) some light is blocked, this destabilizes the radiance distribution in the scene. (c,d) The irradiance at p is approximated by the direct irradiance only. (d) If the light sources are small compared to the scene dimensions the inclusion of a virtual object will cast a hard shadow. The reduction of irradiance for this hard shadow is approximately the same for all points that fall in a shadow volume created by the virtual object (dark grey). This effect can be written as a scaling of the original radiance values to simulate a hard shadow.*

and after the inclusion of the virtual object (d) as:

$$\begin{aligned}
\overline{L}(p,\omega) &= \underbrace{\sum_{p' \in LS} \overline{V_{pp'}} f(p,p') L(p,\omega_{p'}) \cos\theta_{pp'}}_{\text{Direct lighting}} + \underbrace{\sum_{p' \in scene \setminus LS} \overline{V_{pp'}} f(p,p') \overline{L}(p,\omega_{p'}) \cos\theta_{pp'}}_{\text{Indirect lighting}} \\
&= \overline{L_{direct}} + \overline{L_{indirect}} \\
&\approx \overline{L_{direct}}
\end{aligned}$$

The direct lighting terms L_{direct} and $\overline{L_{direct}}$ differ solely due to geometrical reasons, not due to a change in radiance values of the light sources, while the indirect terms differ due to differences in geometry and radiance changes. In this simplified model, the relation between the radiance before and after the inclusion of a virtual object, is a substraction. If the point p receives less light from the direct light sources, it will reflect less light. This can be written as:

$$\overline{L(p,\omega)} = L(p,\omega) - \Delta L \tag{2.24}$$

Often, if the size of the direct light sources is fairly small compared to the scene geometry, the difference in direct lighting is similar (if not identical) for a large set of neighbouring points, which form together a hard shadow. Such a set of points within a hard shadow are shown in dark grey in (d). For the points inside the hard shadow, the amount of direct irradiance is the same. If the BRDF of the points within the

hard shadow is the same, the amount of radiance reflected in direction ω is also approximately the same. In reality, depending on the position of the points inside the hard shadow, an observer might measure a difference in reflected radiance depending on the direction from the scene point to the observer, the direction from the scene point to the light sources, and local BRDF differences.

Under these assumptions, the reduction of the reflected radiance, after the inclusion of the virtual object, can be represented by a scaling factor, which is approximately the same for all scene points with the same BRDF and within the same (hard) shadow. In other words, for all scene points p within the shadow volume shown in (d) and with the same BRDF we can write:

$$\overline{L}(p,\omega) = S \cdot L(p,\omega) \tag{2.25}$$

This implies that for a virtual (hard) shadow, the shadow appearance can be simulated through a scaling factor applied to the original radiance values. If a certain point within a virtual hard shadow already lies inside a real hard shadow, then the scaling as defined in equation 2.25 should not take place at all. Applying a scaling would make that pixel too dark. In a way the virtual scaling of the radiance of such a pixel can be seen as a double scaling: once to simulate the virtual shadow and once due to the original shadow. The geometric scene model can be used to detect which real scene points already lie inside a real shadow. It is obvious, however, that when the geometric model is incorrect, the real shadow estimate will be incorrect too. If the position of the light sources is not known accurately the real shadow estimate will be misaligned with the real shadow in the radiance. Both inaccuracies result in inconsistencies when applying a scaling factor to simulate the virtual shadows.

In the literature several methods exist that simulate common illumination by applying a scaling or substraction on the original radiance values of the points p for which the visibility function has changed [HDH03][SHC+94]. The original radiance values are usually derived from a radiance texture map, or from video footage.

Relighting in general requires a BRDF estimate to simulate the new lighting effects. The BRDF estimate does not need to be complete, knowing the BRDF function for directions towards the observer might be sufficient. Some methods achieve nice results with coarse estimates of the BRDF values in the scene, e.g., [LDR00]. In general better results are expected if the relighting is preceded by inverse illumination, which is explained in section 2.4. Relighting methods also need to detect the position of the real lighting effects such as the shadows, because the relighting method needs to remove these lighting effects prior to applying new lighting effects. Using an incorrect geometric model to detect the lighting effects will lead to artefacts in a similar way as was explained for the common illumination simulation in this section.

2.4 Inverse illumination

The BRDF of a material can be calculated using an image-based sampling approach (e.g., using a goniometer [War92] or using light fields [PD03],[MPDW03],[WGT+05]) or by inverting the radiance equation. In the latter case it is called inverse illumination, and it is this approach that is considered in this thesis. It is called *inverse* illumination as it applies the inverse of the radiance equation to estimate the reflectance from the scene geometry and the scene radiance. An in-depth survey on inverse rendering methods is provided by Patow et al. [PP03]. The following subsections provide more details about how inverse illumination can be achieved for two different types of BRDF models. For both sections the underlying BRDF model assumed is the Ward model [Lar92] which was discussed in section 2.2.2. Section 2.4.1 considers the entire BRDF model including specular effects. In section 2.4.2 this model is restricted to its diffuse component.

The two methodologies described in the following subsections are not exclusive, other methodologies exist. Usually the methods are tailored to fit the exact type of input data available. Following subsections merely serve the task to provide a framework of how inverse illumination can be conducted and to explain how geometry and radiance are related to the inverse illumination process. The following subsections are necessary to understand the concepts, discussions and decisions taken in the subsequent chapters.

2.4.1 Estimating the BRDF of a material

The Ward BRDF model is a complex function of the material's diffuse and specular reflectance and roughness factor. Depending on whether or not anisotropic or isotropic effects are considered the BRDF consist of 3 or 4 parameters. The calculation process of these parameters can be considered as an N-dimensional optimization process, where the optimum is searched for a certain error function. In the case of BRDF optimization, this error function can be the difference between a reference image I and a rendered image R. This image R is rendered from the same viewpoint as I, using the current estimate of the BRDF parameters, the geometric model, and the radiance of all surfaces for which no reflectance is available (including the light sources).

This is illustrated in figure 2.3, where a white box indicates required input data, a grey rounded box stands for a certain operation that takes place on the input data resulting in output data, marked by a black box. The input data are the scene geometry, the scene radiance and some reference images. Usually these reference images are HDRIs and represent scene radiance. First an initial estimate is made for the parameters $P_1, P_2, \ldots, P_N$ of the BRDF model. This estimate is plugged into a rendering system, such as RADIANCE [War94] or PBRT [PH], along with the scene geometry and scene radiance. The scene radiance in this case can be limited to the radiance of the light sources and the radiance of the objects for which no BRDF estimates are made (often called the distant scene). The scene is rendered from the same viewpoints as the reference images, and the difference of the resulting rendered images with the reference images returns an error estimate. This error estimate is used to steer the current BRDF estimate towards the actual BRDF. The iterations continue until a certain convergence criterion is met.

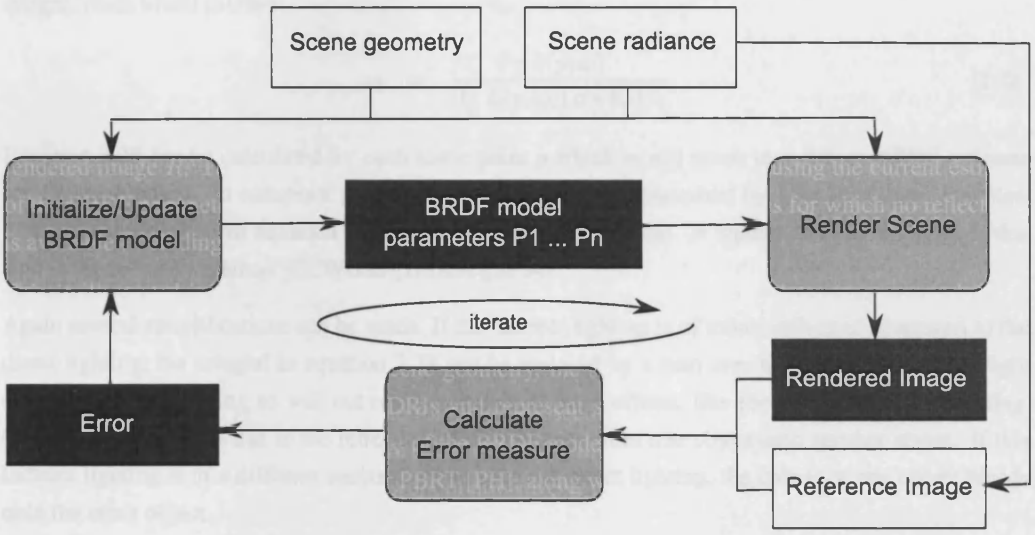


Figure 2.3: *An overview of a typical inverse illumination procedure: a grey box stands for a certain operation that takes place on the input data (white box) resulting in output data (black box). The input data are the scene geometry, the scene radiance and some reference images. Usually the scene radiance is extracted from tthe reference images. The iteration cycle continues until a certain convergence criteria is met.*

Several methods operate similarly to the methodology presented above, e.g., [Deb98][YDMH99][BG01].

Using an iterative approach to estimate or refine the BRDF parameters makes sense as the BRDF model will rarely represent the actual BRDF correctly [NDM05]. The iterations steer the parameters to a BRDF model that fits best with the perceived appearance of the material, though not necessarily with the correct BRDF function.

Different strategies can be used to define the initial BRDF parameters and the update steps. Without going into too many details, Boivin et al. [BG01] start by estimating the diffuse parameter of the BRDF model first; after 1-4 iterations, the model is refined to include specular effects too. Estimating the diffuse BRDF first reduces the search area for the more complex parameters such as the roughness factor.

The importance of knowing the geometry and scene radiance follows immediately from the presented methodology. The scene geometry and radiance are required to create a rendered image that can be compared with a reference image. Small perturbations on the light source positions can create highlights and shadows at locations different from the original scene. As a consequence, a comparison with the reference image can steer the BRDF calculation into an incorrect direction.

2.4.2 Estimating the diffuse material properties

When only considering a diffuse BRDF model derived from the Ward model [Lar92], we can re-write the radiance equation defined in equation 2.18 as:

$$L(p,\omega) = L_e(p,\omega) + \int_\Omega \frac{\rho_d}{\pi} L(p,\omega_i)\cos\theta_i d\omega_i \tag{2.26}$$

Similarly to equation 2.20 we can ignore the emittance L_e:

$$L(p,\omega) = \int_\Omega \frac{\rho_d}{\pi} L(p,\omega_i)\cos\theta_i d\omega_i \tag{2.27}$$

The diffuse component ρ_d is independent of the direction $d\omega$. Therefore it can be written outside of the integral from which follows:

$$\rho_d = \frac{\pi L(p,\omega)}{\int_\Omega L(p,\omega_i)\cos\theta_i d\omega_i} \tag{2.28}$$

Equation 2.28 can be calculated for each scene point p which would result in a diffuse BRDF estimate for all scene points. In computer graphics, a scene is usually represented by a set of discrete samples. Therefore the integral in equation 2.28 can be substituted by a sum. A typical method to calculate this sum is the radiosity method [CCWG88][HSA91][SP94].

Again several simplifications can be made. If the indirect lighting is of minor influence compared to the direct lighting, the integral in equation 2.28 can be replaced by a sum over the radiance from the light sources. However doing so will not remove certain lighting effects, like for instance *colour bleeding*. Colour bleeding exists due to the reflection of the radiance from one object onto another object. If this indirect lighting is of a different spectral colour than the direct lighting, the colour of one object bleeds onto the other object.

It is important to note that the diffuse BRDF is in fact dependent on the wavelength of the light in the scene. This wavelength dependencies is usually ignored or discretized into three components: R, G and B. As discussed in section 2.2.1, radiance and irradiance are also represented by an RGB-triplet. As a result, the BRDF value ρ_d is in fact a triplet $[\rho_{dR}, \rho_{dG}, \rho_{dB}]$. Equation 2.28 can therefore be decomposed into three components, one for each colour component, by representing the radiance L by

its RGB-triplet. Unless explicitly mentioned otherwise, this decomposition into three colour components is implicitly assumed when discussing BRDF values ρ in this thesis.

A real-world scene is rarely entirely diffuse. Nevertheless the diffuse parameter often already gives a good impression of the material. Proof are the methods that obtain a high-quality relit scene when only modelling the diffuse parameter [Deb98][LDR00]. If a scene contains specular effects, estimating the diffuse parameter using equation 2.28 might not immediately return a satisfying diffuse BRDF. Using an iterative approach to estimate the diffuse BRDF using the methodology presented in section 2.4.1 improves the estimate. A suitable update step for the diffuse BRDF of a material point could be:

$$\rho_d^{n+1} = \rho_d^n \times \frac{\Delta I}{\Delta R}, \tag{2.29}$$

where ΔI is the average radiance of that material in the input image, and ΔR is the average radiance of that material in the rendered image. This error update step can be explained intuitively: if the diffuse BRDF ρ_d^n is too small, the rendering of the material is too dark, making the ratio $\frac{\Delta I}{\Delta R}$ larger than 1. Hence, the updated diffuse BRDF ρ_d^{n+1} is larger than its previous value ρ_d^n. Such an update method is unstable when the ratio $\frac{\Delta I}{\Delta R}$ drives the diffuse BRDF value ρ_d towards a value larger than one. This might occur when the estimation of the radiance of the light sources used during the rendering is incorrect, and lower than the actual radiance of the light sources.

From the methodology described above to estimate the diffuse material properties of an object, we can see the importance of knowing the scene radiance and the scene geometry. When estimating the diffuse reflectance of point p using equation 2.28, the radiance of p and of all scene points visible from p need to be available. Usually the radiance of these scene points is derived from HDRIs. The scene radiance can be derived from more than one HDRI, as long as all radiance values retrieved result from the same system of radiance equations that defines the scene radiance for point p in the first place. If the scene radiance is captured from more than one image, and the scene illumination is dynamic throughout the HDRI capture, the scene radiance distribution cannot be reconstructed for a certain scene point p. This poses several problems, which are further explained and resolved in chapter 6.

The geometric model is necessary to derive the angles $\cos\theta_i$. From equation 2.28 we can deduct that an error on the geometry for points with a large radiance value has a bigger influence on the inverse illumination calculation than inaccuracy in the geometry of points with a low radiance value. In other words, it is important to correctly model the positions of the light sources in the scene.

Chapter 6 presents a relighting system that calculates its reflectance values in a similar manner as explained in this section. That same chapter provides more practical considerations about the inverse illumination implementation, such as how to derive the scene illumination, how to render the images R, and if the BRDF should be calculated per triangle or on a per-pixel basis.

2.5 High dynamic range imaging

High dynamic range images (HDRIs) are used to capture the radiance in the scene. High dynamic range imaging is a fairly new discipline, developed to compensate for the under-performance of the camera hardware, such as the saturation that occurs after a too long exposure time. The book from Reinhard et al. [RWPD05] gives an excellent overview of the existing methods to generate and display high dynamic range images. This book also addresses the encoding issues, as high dynamic range images represent a huge amount of data compared to conventional images.

This section gives a brief overview of high dynamic range imaging, focusing on the generation, rather

than on the display or encoding. Additional information, less relevant to this thesis, can be found in [RWPD05]. Section 2.5.1 defines the concept high dynamic range images. Section 2.5.2 explains how the generation of high dynamic range images usually proceeds. Section 2.5.3 discusses the limitations of the current generation methods in terms of robustness against uncontrollable situations. Section 2.5.4 describes briefly how a HDRI can be viewed on a screen.

2.5.1 High dynamic range imaging: motivation and definition

HDR imaging refers to the set of techniques used to represent images with a greater dynamic range than conventional imaging methods allow. The definition of *dynamic range* depends on the context in which it is used. One common definition is the ratio between the largest and the lowest representation of a certain physical measure, usually represented in dB. From this we can define the dynamic range, depending on the context in which it is used, as:

- *Scene*: the ratio between the brightest and darkest parts of the scene.
- *Analogue camera*: the ratio between the highest and lowest luminance value that is captured with the camera's sensor without saturation and above a certain signal to noise ratio. In analogue imaging, the camera's sensor consists of film, which changes in constitution after exposure to light.
- *Digital camera*: the ratio of the maximum luminance that receives a unique coded representation (the *saturation* luminance) to the lowest luminance for which the signal to noise ratio is 1.0, according to the ISO standard [Intb]. The dynamic range of a camera depends on the specifications of its sensors. In digital imaging, the sensor is usually a CMOS or a CCD, which receives a certain voltage after exposure to light that is further digitized to an intensity value.
- *Display device*: the ratio between the maximum and minimum amount of emitted light.

Pixel *saturation* occurs when the camera's pixel sensors are pushed towards their highest representable values. In an image a pixel is saturated if any of its RGB components has a value of 255. A fully saturated pixel is represented as a white RGB triplet: $[255, 255, 255]$. A pixel is called *under-exposed* when its value is affected by image noise and therefore unreliable.

The *tonal range* is another concept often linked to a camera and an image. In general the tonal range is the total number of values that can be represented by a signal. For a camera it is the number of different intensity values used to represent a pixel. When a digital camera's sensor is exposed to light, it receives a value (voltage) depending on the amount of light that reaches the sensor over the time interval of the exposure. When these sensor values are read out, they are quantized. The number of quantization levels used defines the tonal range of the camera.

Some typical luminance values of a real-world scene (Wandell et al. [Wan95]) are given in figure 2.4. The dynamic range of an outdoor scene can easily have a ratio of $10^8 : 1$. Often the dynamic range is given as a log_{10} ratio, in that case the previous ratio is given as an order of 8.

The dynamic range of film is usually derived from the *characteristic curve*[1], first described by Hurter and Driffield [HD90] in 1890, and for film measured through plotting the (log_{10} of the) film exposure versus the density of the film. The density of film is given by the concentration of dyes or silver salts remaining on the film after exposure to light. A high density relates to a dark image (short exposure), while a

[1] This is also often referred to by the sensitometric curve, the D Log H (or E) curve, or the H&D (Hurter and Driffield) curve.

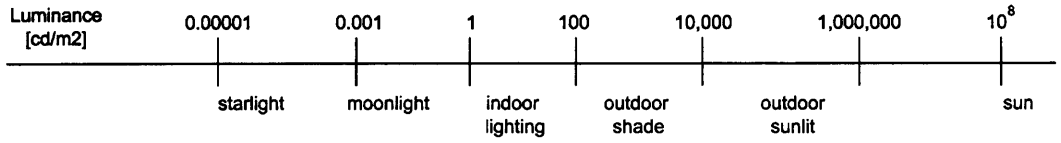


Figure 2.4: *Luminance values for a typical indoor/outdoor scene [Wan95].*

low density relates to a bright image (long exposure). Figure 2.5 shows a characteristic curve of film for each colour emulsion. The curves have an S-shape and the figure illustrates that film emulsions can cover about 4 orders. The characteristic curves differ for different types of film, and depend on storage, handling, manufacturing and processing conditions. Digital cameras typically have a lower dynamic range than film or analogue cameras. An order of only 2 is a common value [RWPD05]. This means that no conventional camera (analogue nor digital) can capture a scene containing a bright sun in the sky and dark shadows without showing saturation or under-exposure, as according to figure 2.4 such a scene would extend a dynamic range of approximately $(\frac{10^8}{10^3} =)10^5 : 1$ or 5 orders.

An HDRI is an image whose pixels are proportional to the scene radiance, and do not contain any saturation or under-exposure. Usually the pixels of an HDRI are stored as 32-bit floating point numbers per colour channel. Conventional *Low Dynamic Range Images (LDRI)*, are represented by the traditional 8-bit per channel, or 16-bit per channel (RAW image format) integer images. Another difference between an HDRI and an LDRI is the linear relation between the pixel values of an HDRI and the (ir)radiance in the scene, while an LDRI usually represents intensity values, derived from the irradiance. Appendix B lists some standard HDRI formats and their encoding schemes. The dynamic range represented by these formats is listed in table B.2.

2.5.2 High dynamic range imaging: generation

Several methods exist to generate HDRIs. An overview is given in [NB03]. The most low-cost solution calculates the HDRI as a composition of a set of LDRIs captured with different exposures. Other HDRI methods adapt the functionality of conventional cameras to capture HDRIs. This thesis focuses on the first type of HDRI generation. The purpose of this section is to explain the concept of multiple exposures in HDRI generation, to give a theoretical overview of HDRI generation using multiple exposures, to provide an example, and to discuss the advances made in HDR cameras.

A Multiple exposures

The camera curve defines the relation between the scene irradiance (hitting the camera sensors) and the intensity that we see in the image. The camera curve depends on several settings, such as the exposure. The exposure e of a camera is the product of the exposure time T with the aperture width d used to capture the image, or $e = T \cdot d$ with:

- T: The exposure time or shutter speed used to expose the camera sensor. Usually a camera allows the user to select some pre-fixed values such as: 8", 4", ..., $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, ... ,$\frac{1}{4000}$.
- d: The aperture or f-stop, which defines the opening of the lens. Usually a camera device allows the user to select some pre-fixed values such as: f16, f11, f8, ..., f2.8

Choosing a larger exposure time or larger aperture width will push the pixel sensors into saturation; the image is brighter than when a lower exposure time is chosen. This observation lies at the basis of HDRI

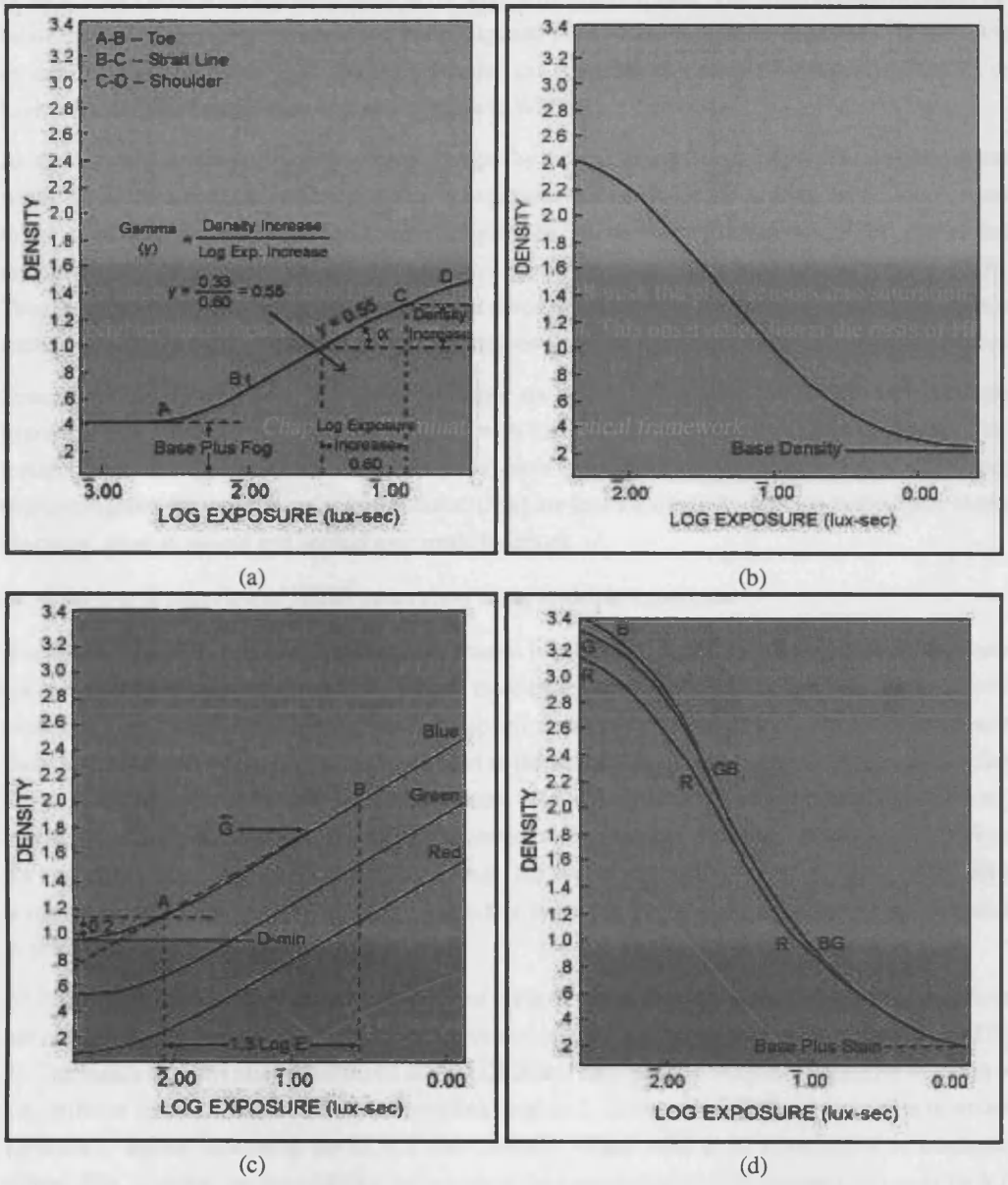


Figure 2.5: *The sensitometric or characteristic curve for black and white negative (a), black and white reverse (b), colour negative (c), and colour reverse (d) film. The characteristic curve of a film emulsion can extend about 4 (log_{10}) orders of magnitude, this is a ratio of 10.000:1. Courtesy © Eastman Kodak Company.*

generation using multiple exposures. When a set of LDRIs are captured with varying exposure settings (simply referred to from here on as *exposures*), at the same time and from the same viewpoint, these exposures can be combined into one HDRI [MP95][DM97][MN99][GN03a]. First the camera curve is used to transform the pixels in the exposures to irradiance values. Then the final HDRI is generated as a per-pixel weighted average of the irradiance values in the exposures. The main difference between existing HDRI generation algorithms lies in the weighting function used. The weights need to reflect the following underlying idea: saturated and under-exposed pixel values need to be suppressed as they give an incorrect radiance value. The weighting function can therefore be a simple hat-function [DM97], or have a more sophisticated shape as is described in [RWPD05].

As said earlier, the exposure settings can be changed by either altering the exposure time, or the aperture width. Thus, the maximum number of different exposures that can be captured with one camera is equal to the number of different exposure times multiplied by the number of different aperture widths that are allowed by the camera. Usually, between $3 - 11$ different exposures are captured for one HDRI. Therefore it often suffices to change the exposure speed while keeping the aperture width fixed. This is further encouraged by the undesired change in depth-of-field that a change in aperture width introduces.

Several commercial and non-commercial packages are available that generate HDRIs from multiple exposures in a relatively painless manner, often with limited user-interaction. HDRshop [HDR], Photomatix [Pho] and Adobe's Photoshop CS2 [Ado] have a user friendly graphical interface. The open source programs Photosphere [Any] and Rascal [Ras] are less user-friendly but due to the open source character, allow to amend and append user-made functions.

B Theoretical overview of HDRI generation using multiple exposures

A schematic overview of the HDRI generation process is outlined in figure 2.6. First a set of N exposures are captured. In the example shown $N = 3$ and three exposures I_i with $i \in \{1, 2, 3\}$ are captured, with exposure times T_i with $i \in \{1, 2, 3\}$, while the aperture width is kept the same. Some of the exposures show saturation, their intensity values are clipped to the maximum intensity value (e.g, the circular disk in I_3). This is due to the fact that the exposure time used to capture I_3 (T_3 in our example) is too long, causing the pixel sensors to saturate. Other exposures can show under-exposure, the pixel intensities do not exceed the noise level of the pixel sensors (e.g., the background in I_1). To create a correct HDRI it is important to choose the exposure settings such that every part of the scene appears without saturation in at least one of the different exposures.

An HDRI H is generated as a weighted average of the different exposures, where the weights are defined per pixel and per exposure. This is achieved through constructing a weighting mask W_i for each LDRI I_i. The masks have the same dimensions as the LDRIs and each pixel in W_i is a measure for the level of saturation or under-exposure of the corresponding pixel in I_i (lower values indicate saturation or under-exposure). Before combining the LDRIs their intensity values need to be transformed to irradiance values. This is carried out by applying the inverse of the camera curve on the intensity values in each I_i to create an irradiance image E_i.

The camera curve defines the relation between pixel intensity and image sensor irradiance and is defined by all the processes that are involved during this transition. The sensor's (film, CMOS or CCD) response curve is one of the contributors to the camera curve. The curve can be measured using a Macbeth colour chart [CR96] or can be estimated from the same set of exposures used to generate the HDRI [MP95][DM97][MN99]. The camera curve can be very different between cameras and but is in general non-linear. Interestingly enough, the camera curve of a digital camera is often digitally tuned to mimic the non-linear behaviour of film. In [GN03b] over 200 different real-world camera curves are collected

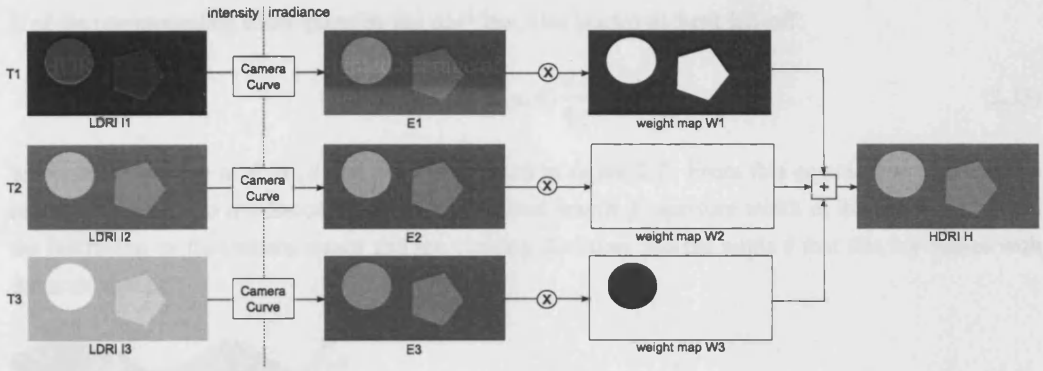


Figure 2.6: *HDRI generation: a theoretical overview. A set of N (=3) different LDRIs I_i are captured with different exposure times T_i. The camera curve is used to transform these exposures to irradiance images E_i. The final HDRI H is calculated as a weighted average of the E_is, where the weights are defined by the saturation of a pixel in its original LDRI I_i. In the figure shown, the weights are stored in a weight map W_i, the weights can have any value between [0,1].*

in a database. From this database it can be derived that the camera curves are in general monotonic and increasing.

Following the same strategy as Debevec et al. [DM97], the transition from camera irradiance to pixel intensity using a camera curve $f(\cdot)$ can be written as:

$$I_i(k,l) = f(E_i(k,l)T_i) \tag{2.30}$$

where (k,l) are the pixel coordinates and I_i, E_i, and T_i respectively the input image, irradiance image, and exposure time for exposure i as is illustrated in image 2.6. If we assume the camera curve is monotonic, $f(\cdot)$ can be inverted to $g(\cdot)$ and we can write:

$$E_i(k,l) = \frac{g(I_i(k,l))}{T_i} \tag{2.31}$$

The final pixels in the HDRI H are generated as follows:

$$H(k,l) = \frac{\sum_{i=0}^{N-1} W_i(k,l)E_i(k,l)}{\sum_{i=0}^{N-1} W_i(k,l)} \tag{2.32}$$

with N equal to the number of exposures used. As mentioned previously, the weights need to reflect the idea that saturated or under-exposed pixels are unsuitable to represent irradiance, or, that these pixels should be withheld from the actual combining into an HDRI. An in depth-analysis into the choice of weighting function is given in [RWPD05]. An advantage of the merging of corresponding pixels to an irradiance value is that it ensures that glare and image noise (due to, for instance, lens imperfections) that might be present in an LDRI is reduced in the final HDRI [Deb04][RWPD05].

Before continuing our discussion about HDRIs, it is important to explain the relation between radiance, irradiance and the pixels in an HDRI. The camera curve defines the relation between a pixel's intensity $I(k,l)$ and the irradiance $E(k,l)$ that fell on the camera sensor. This irradiance is related to the radiance

L of the corresponding scene point by the cos^4 law, also known as light fall-off:

$$E(p,\theta) = L(p,\theta)\frac{\pi}{4}\frac{d^2}{f^2}cos^4\alpha \tag{2.33}$$

where the definition of θ, α, d and f are illustrated in figure 2.7. From this equation we see that the radiance is related to irradiance by the camera's focal length f, aperture width d, the angle α between the ray falling on the camera sensor and the viewing direction, and the angle θ that this ray makes with the normal at p.

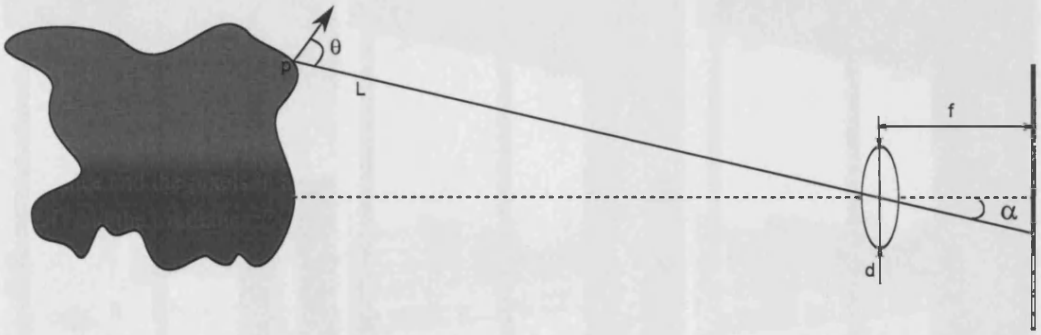


Figure 2.7: *Light fall-off: from irradiance to radiance.*

This fall-off, also known as optical vignetting, reduces the brightness of a pixel the further away from the centre of the lens. Most modern cameras have been designed to remove this relation to the angle α [DM97][KMH95], therefore an HDRI is often referred to as a radiance image. In this thesis we correctly refer to irradiance values when discussing the physical interpretation of the pixels in an HDRI, unless mentioned otherwise.

The influence of optical vignetting is larger the longer the lens or the smaller the focal length, and the larger the aperture width used. The dependence on α makes this reduction dependent on the position of a pixel in an image. As an illustration, the quantity $cos^4\alpha$ for a lens with a diagonal field of view of 6.2 degrees[2] is equal to 0.9965 at its worse point (α equal to 3.1 degrees), which is less than 1% reduction of the original radiance value.

C An example

An example of an LDRI exposure sequence is given in figure 2.8. The six exposures are captured from the same viewpoint and using different exposure settings. The scene contains a considerable dynamic range: the sky is bright, the inside of the elevator is dark. None of the six exposures captures both these parts without showing saturation or under-exposure. Figure 2.9 (a) and (b) illustrates the resulting HDRI (using different tonemapping schemes), generated by combining the six LDRIs shown in figure 2.8. The HDRI succeeds in representing the sky, indoor lights and inside of the elevator without saturation. The HDRIs in figure 2.9 are *tonemapped*, which is an image operation that skews and quantizes the floating point pixel values such that they are represented by 8-bit per colour channel. More details about tonemapping are given in section 2.5.4.

D High dynamic range cameras

As was discussed in section 2.5.1, the dynamic range of a camera is the ratio of the maximum over the minimum incoming light that can be captured without showing saturation or under-exposure. Some

[2]This is the field of view of the Sigma AF 135-400mm f/4.5-5.6 APO Aspherical RF, when set to 400mm.

Figure 2.8: *HDRI generation to increase the dynamic range of an image: these six images are LDRIs captured from the same viewpoint, but with a different exposure setting. The difference in exposure speed is 1-fstop. The scene contains considerable dynamic range: the sky is bright, the inside of the elevator is dark. None of the images captures both those parts without showing saturation or under-exposure. Figure 2.9 shows two HDRIs generated using the LDRIs shown here, but displayed using a different tonemapping operation.*

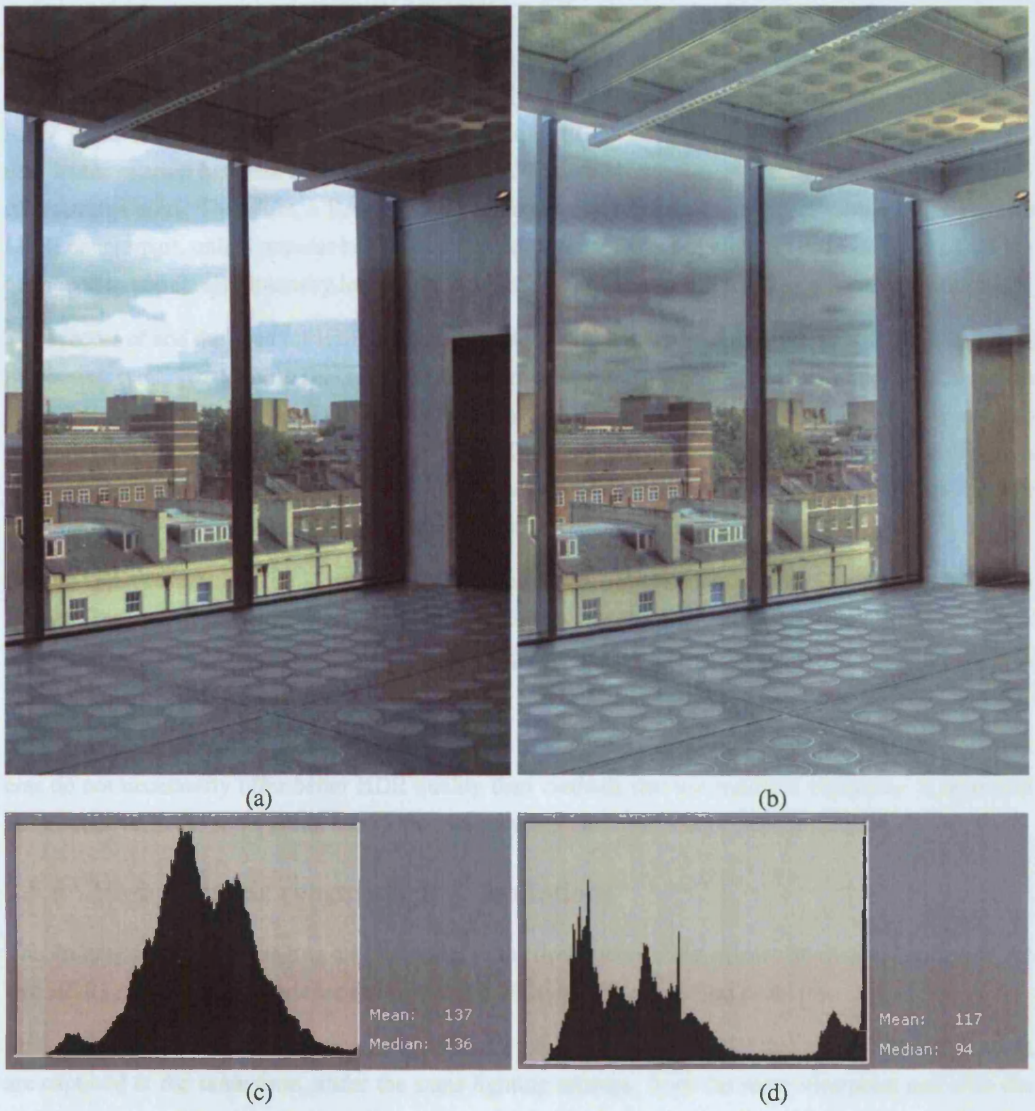


Figure 2.9: *Both images (a) and (b) are tonemapped HDRIs generated from the exposure sequence shown in figure 2.8. The difference between the two HDRIs is the tonemapping scheme. (a) is created by rounding the floating point values in the HDRI to the nearest integer within [0,255]. (b) uses the tonemapper from Photomatix [Pho]. The HDRI shown in (b) succeeds in representing the sky, indoor lights and inside of the elevator without saturation. (c) and (d) illustrate the histogram of (a) and (b) respectively. The histogram shown in (d) is more equalized than that shown in (c).*

digital cameras can export images in the *RAW image format.* A RAW image, represents the image data with minimal processing, each pixel value corresponds to the sensor's value, represented by 12 or 14 bit per channel instead of the conventional 8 bit for JPEG images. Usually an image is read out as an LDRI, where the intensity (I) of a pixel is related to its RAW value (R) as follows:

$$I = D(R^{\frac{1}{\gamma}}) \cong exposure^{\frac{1}{\gamma}} \tag{2.34}$$

where γ defines the gamma correction, which is an internal adjustment to compensate non-linear characteristics of devices like screens and cameras, and $D(\cdot)$ is a non-linear post-processing function. The RAW value R from equation 2.34 is the direct output of the camera's sensors, usually there exists a near-linear relation between camera sensor and RAW value. Nevertheless it still depends on the exposure settings used. Therefore, a RAW image does not really represent a higher dynamic range than its LDRI counterpart, unlike popular belief. The main difference is that the intensity values in an LDRI are quantized to 256 integer intensity levels, while a RAW image is usually stored as a floating point image.

The success of and the need for HDRIs have encouraged the development of cameras with built-in HDRI processing (e.g., [BAS][NB03][NBB04]). Even an extension to MPEG video is under consideration (e.g., [MKMS04]). The type of HDR camera technology varies [NB03]. Some apply spatially varying neutral density filters (e.g., [AA01]) or split the LDRI optically using beam splitters to recombine the different images at a later stage (e.g., [Sai95]). Others use a technology called adaptive dynamic range imaging, in which the pixels of a camera are exposed at different times (e.g., [NB03]).

An example of a commercially available HDR camera is the SMAL Ultra Pocket camera from SMAL Camera Technologies [SMa] which captures HDRIs at a resolution of 482×642. Another example is the SpheronCam HDR from SpheronVR [Sph], a high resolution, high-quality HDR camera, which uses a line scanner to scan the entire environment.

HDR cameras are expensive or rely on technologies that are not commercially available yet. These cameras do not necessarily offer better HDR quality than methods that use multiple exposures to represent HDRI content of the same scene due to their lengthy capture time or their poor resolution.

2.5.3 High dynamic range imaging: limitations

The limitations of HDR imaging originally lies in the limitations of the camera hardware. Neither of the two HDRI generation methods are error-free, and both suffer from similar problems.

First of all, for the HDRI generation using multiple exposures it is essential that the different exposures are captured at the same time, under the same lighting settings, from the same viewpoint and with the same real scene object constitution. However, a conventional camera usually needs to capture the exposures sequentially, where the smallest time lapse possible is equal to the sum of the exposure times used. In practice the time lapse of such a sequence is longer, especially when the exposure times need to be set manually. The scene must therefore not change during the capturing. In dynamic environments, like cloudy, windy outdoor scenes, or scenes which contain moving objects like people and cars, conventional HDRI generation methods using multiple exposures fail. Also, the requirement that the camera cannot be moved limits the usability of the generation method, as it implies carrying around a tripod, whenever an HDRI needs to be taken.

HDR cameras suffer from similar problems. When they scan the scene in lines as the SpheronCam [Sph], the lengthy capture time will introduce similar problems. Hence, object deformation, illumination changes and camera movement are also sources of errors for this type of HDRI generation. Other

problems are the low resolution of the camera [SMa]. However, foremost the main problem with the HDR cameras is that they are expensive or not commercially available. This encouraged us to focus on developing an improved HDRI generation method for uncontrollable environments using multiple exposures, rather than relying on the HDR cameras.

In chapter 5 a more detailed overview is given of the errors introduced by camera and object movement when using multiple exposures to generate the HDRI. In that same chapter a solution is presented that allows limited object movement, and allows the images to be captured with a hand-held camera.

2.5.4 High dynamic range display

Display devices are tuned to represent the colour channels with only 8-bits, as this corresponds to the 8-bit per channel of image data and video. This presents the problem of how to represent a floating point image, or video, without saturation. The process of skewing and processing of HDR data in order to be represented with 8-bits per channel is known as *tonemapping*. Various different tonemapping schemes exist. Often they use the physiology of the human eye to best represent the floating point values with only 24 bits.

A tonemapping algorithm either works globally or locally. Globally means that it scales all pixels according to the same global curve. Local algorithms, find the best scaling for a pixel based on the pixel values in a local region around the pixel. The latter type of algorithms often focus on improving the contrast in an image. Popular tonemapping algorithms are [TR93][WRP97][PTYG00][RSSF02], but this list is far from being exhaustive. A good overview of the state of the art in tone reproduction is given by Devlin et al. [DCWP02]. An interesting evaluation of different tonemapping schemes is presented by Ledda et al. [LCTS05].

Figure 2.9 illustrates the difference in two HDRIs generated in exactly the same manner, but tonemapped using different tonemapping schemes. The luminance values of the HDRI are scaled to the nearest integer in (a), while a more sophisticated tonemapping scheme[3] is used in (b). The histograms of the images (a) and (b) are shown in (c) and (d) respectively. The histogram in (c) is skewed compared to that of (d) which is much flatter. Unlike the intensity values in (a), the intensity values in (b) are not proportional to the irradiance values in the scene. Tonemapped images can therefore often look unrealistic. In this thesis the HDRIs are always non-linearly tonemapped, unless mentioned otherwise.

With HDRI techniques gaining popularity, the first HDR displays are being developed. Brightside Technologies recently presented the DR37-P [Bri], a second generation HDR display based on the methodologies presented by Seetzen et al. [SHS+04].

2.6 Chapter summary

Basic radiometric concepts such as radiance, irradiance, and reflectance have been explained in this chapter. These concepts form the basis for the radiance equation, which defines the radiance of a scene point into a certain direction, based on its BRDF, the scene geometry, and the radiance of other scene points. After inclusion of a virtual object, or changing the light sources, the radiance equation needs to be re-calculated to retrieve a new scene radiance distribution. Illumination methods re-calculate the radiance equations for the scene points, either correctly, or through a set of approximations.

Some illumination methods require the knowledge of the BRDF of the materials in the scene. The process of retrieving the parameters of a BRDF model from the scene radiance and geometry is called

[3] To generate (b) the build-in tonemapping scheme of Photomatix [Pho] was used.

inverse illumination as it applies the inverse of the radiance equation onto the geometry and radiance. In this chapter a methodology is presented that can be used to retrieve the parameters of a certain BRDF model using an iterative process, where the parameters of the BRDF model are updated using an error estimate. This error estimate is extracted from the difference between a rendered image and a reference image.

In this chapter it was argued that illumination methods for mixed reality require the scene geometry and scene radiance as input. All illumination methods are sensitive to inaccuracies in the radiance and geometry capture. If the radiance and geometry cannot be accurately captured, inaccuracies in the simulated virtual illumination are expected, unless the illumination methods take special care to prevent such inaccuracies.

Finally, this thesis also provided an overview of high dynamic range imaging, as these methods are used to extract the scene radiance. The underlying theory of HDRI generation along with how HDRIs are displayed were explained. To generate an HDRI using multiple exposures several conditions need to be obeyed: the scene geometry and illumination needs to be static, no camera movements are allowed. These are fairly restrictive conditions, which reduce the applicability and robustness of the HDRI generation process in uncontrollable scenes.

Chapter 3

Review of illumination methods for mixed reality

3.1 Introduction

This chapter provides a definition of mixed reality and gives an overview of the most relevant illumination methods that are available in the literature. In short, a mixed reality is an environment containing both real and virtual elements. The illumination methods attempt to simulate the (virtual) lighting effects induced by the inclusion of the virtual object in the real scene, and can be categorized into three groups: common illumination, relighting and physically-based illumination. All three types make the illumination between the virtual and real objects consistent, but differ in how this consistency is obtained.

The three methods require a model of the geometry and the radiance of the 3D scene as input. The accuracy at which these two inputs need to be known differs among the methods. Gathering the input data is subject to many errors, and can be very laborious. This encouraged us to classify the existing illumination methods into different groups, based on the amount of input data needed, rather than according to the type of illumination obtained. Four different groups are analysed, where each group requires a different amount of input data to enable illumination simulation. Assessing other characteristics such as computation time, flexibility and quality delivered, reveals that methods within a group often share the same characteristics. This is somewhat expected as it seems obvious that the more input data is required, the better the quality of the simulated illumination obtained, but that in general more processing time is required. The presented classification and the conclusions drawn have been presented as a State of the Art (STAR) report at Eurographics 2004 [JL04] and a slightly amended version has been published in Computer Graphics Forum [JL06].

This chapter is organized as follows. Section 3.2 defines the concept of mixed reality and gives more information about illumination methods for mixed reality and how these methods can be assessed and compared. Section 3.3 presents a classification of these methods into four different groups. Section 3.4 gives a brief discussion and comparison of the illumination methods listed in the previous section. Once the classification is presented it is possible to position the methods presented in this thesis within the illumination framework, as is explained in section 3.5. Finally section 3.6 gives a summary of this chapter.

3.2 Mixed reality

This section provides a definition of mixed reality, see section 3.2.1, and familiarizes the reader with the types of illumination methods available in the literature, see section 3.2.2. Section 3.2.3 discusses the common requirements of the three types of illumination methods, and explains how the different types of illumination methods can be assessed and compared based on a few identifiers.

3.2.1 Mixed reality: a definition

To understand the concept *mixed reality* it is necessary to classify the different types of environments that can be generated with a computer. Milgram et al. [MK94][OT99] present such classification based on the amount and type of virtual and real elements that constitute the resulting world. In their classification, all possible environments form one continuum called the *reality-virtuality (RV) continuum*, see figure 3.1. In this continuum, four worlds can be identified that have an outspoken character. These four worlds define four ranges in the RV continuum; between some of these worlds there are no well-defined boundaries. The first and most straightforward of these is the real world without any addition of virtual elements; it is referred to as *reality* and it lies on the left end of the RV continuum. In the second environment, virtual elements are added to the real world. This world is referred to by the term *augmented reality (AR)* [Azu95][ABB+01][BKM99]. In an opposite scenario, the world consists of a virtual environment, augmented with real elements. This world is consequently called an *augmented virtuality (AV)*. The last environment, on the right hand side, does not contain any real elements and is therefore labelled as a *virtual environment (VE)*. The term *mixed reality (MR)* refers to those worlds that are a mix of virtual and real elements, or, MR spans the RV continuum. Illumination methods for MR can operate in real-time or not. When an AR method is specifically designed to work in real-time it is preferred to refer to the method with the term AR, rather than with the generalized term MR. This chapter discusses the various existing illumination methods for MR applications in general, therefore the term MR is often preferred to AR or AV separately. However, whenever the focus lies on the real-time character of the application, the expression AR is used.

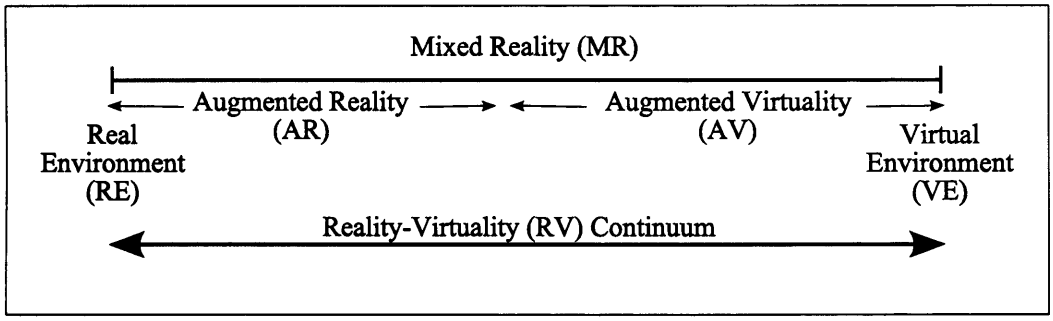
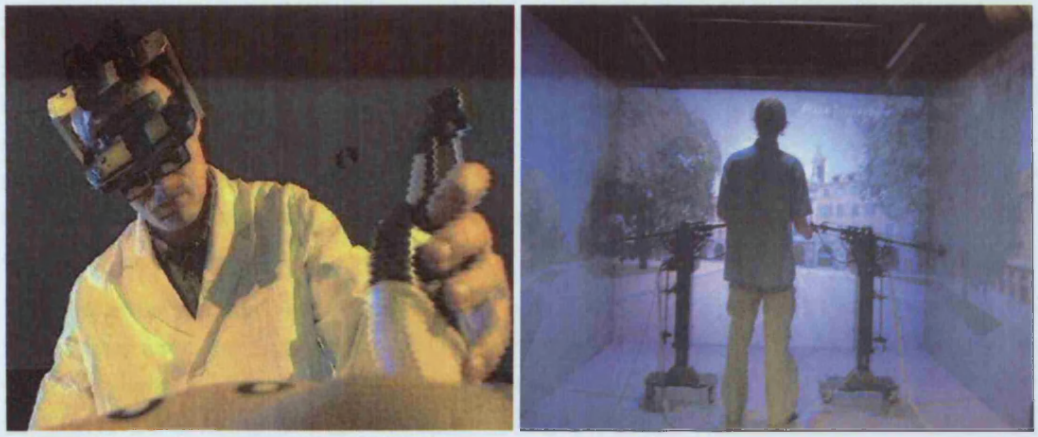


Figure 3.1: *Simplified representation of the* Reality-Virtuality Continuum*[MK94][OT99]. This continuum can be divided into four main fields, based on the proportion of real/virtual objects present: Reality, Augmented Reality, Augmented Virtuality and Virtual Reality.*

Two different classes of AR exist; these classes differ in the realization of the AR [MK94]. The first class groups the methods for semi-transparent or see-through displays, examples are given in [SCT+94][BGWK03]. Two different see-through display methods exist. The first method (optical AR method [Azu95]) projects the virtual objects on a transparent background, most likely the glasses of goggles, see figure 3.2 (a). The second method (video AR method [Azu95]) uses a head-mounted display: a head-mounted camera records the environment and projects the recording inside the display together



(a) (b)

Figure 3.2: *Two different types of realisation of AR. (a) See-through display. Courtesy of Siconet-AR, Zentrum fur Graphische Datenverarbeitung. (b) Computer Augmented Reality: The CREATE [LWR+03] project at UCL.*

with the virtual objects. The second class of AR replaces the expensive see-through devices with a non-immersive display; it is usually called a *computer augmented reality (CAR)*[DRB97], see figure 3.2 (b). CAR often uses a predefined 3D model with textures mapped onto it to represent the real scene, and does not necessarily show live video footage. The immersion of the first class is more likely to be higher than that of the second class, as the user will be slightly more distant from the scene in the latter case. On the other hand, see-through AR is more complicated to implement than CAR. Fortunately, see-through devices are not always required by the application: urban planning, architecture and some applications in the entertainment industry are satisfied with the second type of CAR display methods.

In earlier approaches of AR, virtual objects were positioned on top of a real environment. Calibration and registration are difficult processes and initially the focus lay upon taking into account the possible occlusion and collision effects, while no further adaptations on real and virtual objects were carried out. In other words, after the inclusion no resulting shadows were generated nor were new illumination settings applied to the merged scene. This type of research led to the development of algorithms to track a camera in a scene. For instance, the software library ARToolkit [KBBM99] uses fiducials in the scene to track the camera position in real-time. Simultaneous Localization and Mapping (SLAM), is an other research field which focuses in part on real-time automatic camera localisation and automatic landmark reconstruction, without the use of externally included fiducials. VSLAM (visual SLAM) does this based on visual input (such as camera footage) and is therefore perfectly suitable for AR applications. Recently Davison et al. [Dav03] presented SLAM using a low-cost camera, which could enable low-cost AR in an originally uncalibrated environment.

AR systems that do not attempt to simulate the lighting effects between virtual and real objects do not yield a high level of realism, as consistency between the objects is restricted to geometric aspects. To improve the amount of realism, illumination methods have been developed. Three illumination methods can be identified that attempt to raise the quality of AR and in general MR: common illumination, relighting and physically-based illumination. A brief overview of these methods was already given in chapters 1 and 2 and it was explained that these methods vary in the type and quality of the lighting effects obtained. The next section illustrates these types of illumination methods based on some examples.

At the moment, good illumination methods exist that can relight an augmented scene with a different type of scene illumination. It is getting more difficult to differentiate between virtual objects and real

objects. The main limitation of most methods is the lengthy pre-processing time and the slow update rate, which excludes real-time applications. When a geometric model of the scene is required, the user has to create one, usually in a semi-manual and error-prone manner. The scene update rate is often too slow to allow real-time user interaction, even with the current progress in computer hardware and software. The research focus is moving towards using hardware for the calculation instead of software to accelerate computation. Early results are promising, but more research needs to be carried out in this area.

This chapter does not review all existing work. Instead it concentrates on illumination methods for MR that are meant for large environments. When optimized and extended, these methods can be widely applicable in real-time applications, for instance see-through display in AR. Several techniques exist for relighting human faces [Mar98][DHT+00], or that focus on local objects or simple scenes [ALCS03][SWI97]. These techniques are classified mainly in the domain of inverse illumination as the emphasis was placed on this aspect in the referenced papers. Although these techniques are designed for small objects they can be used to build useful and strong methods for illumination in MR but they are not further discussed in this chapter.

3.2.2 Illumination methods for mixed reality

The various existing illumination methods can be grouped into three different classes, based on the methodology used to tackle the problem. They are already listed in the introduction and are further discussed in this section:

A Common illumination

To this category belong all methods that provide a certain level of illumination blending, like the addition of shadows projected from real objects onto virtual objects and shadows cast by virtual objects onto real objects. These methods do not allow any modification of the current illumination of the scene. Two different types of common illumination can be considered: *local* and *global* common illumination, referring to the type of illumination simulated. For local common illumination, there is usually no requirement of any BRDF information. For global illumination, it is often important to have an estimate of the material properties of the real objects. The accuracy of these types of methods depends on the accuracy of the known geometric model of the real scene. Figure 3.3 gives an example of a rendering using global common illumination [SSI99]. In this example, the shadows of the virtual object (glossy dodecahedron (top right), specular sphere (bottom right)) are generated using scaling of the underlying texture, the illumination from the scene onto the virtual object is simulated as well, this is mainly visible through the reflection of the scene in the sphere.

Examples of applications that use common illumination to improve the MR can be found in the movie industry. Special effects in movies make an effort to mix lighting effects and reflections as realistically as possible, resulting in brilliant graphical effects in recent movies such as *Jurassic Park*, *Harry Potter* and *The Lord of the Rings* trilogy. In these movies computer-generated effects are blended entirely with the real footage; usually this is carried out by hand.

B Relighting after light removal

Relighting methods make it possible to change the illumination of the scene in two steps. Firstly, the current lighting effects of the real scene are analysed and possibly removed. Secondly, new lighting effects (shadows, intensity changes, addition of a new light, indirect lighting effects, etc.) are generated based on a new illumination pattern. These methods do not necessarily require an exact knowledge of

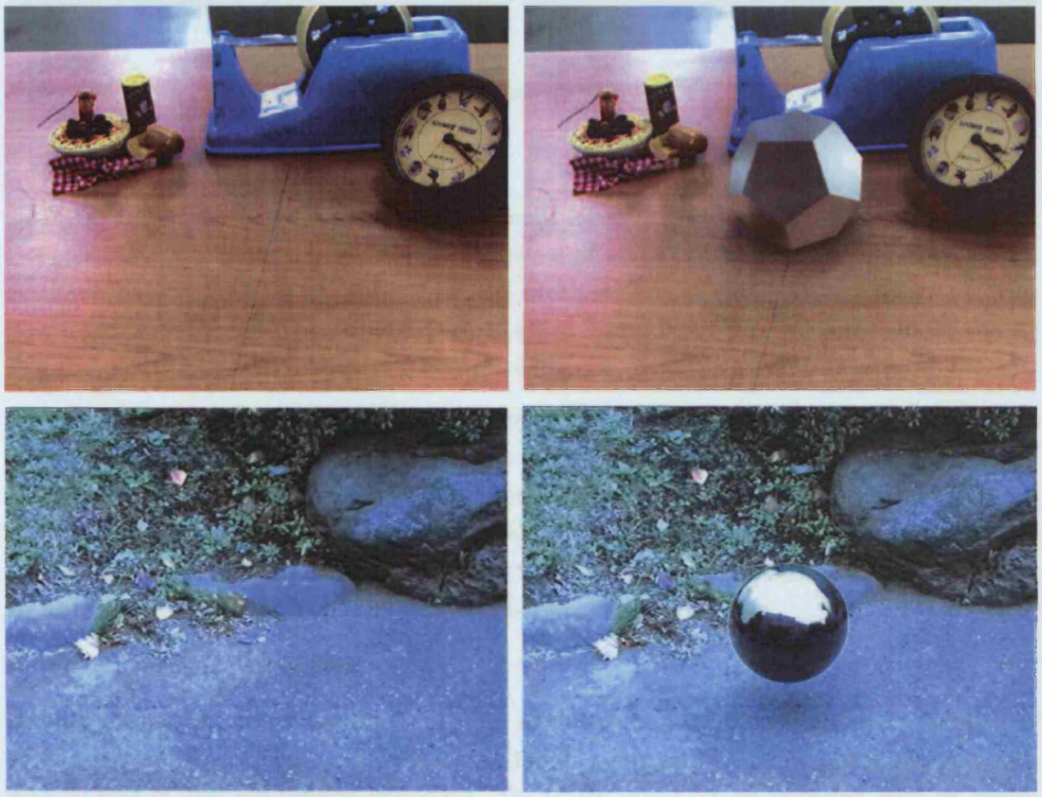


Figure 3.3: *Results for Sato et al. [SSI99]. The top row shows results for an indoor scene, the bottom row for an outdoor scene. The images on the left are the input images, the images on the right illustrate the resulting MR. Soft shadows are produced using local common illumination. Courtesy of Sato et al.*

(a) (b)

Figure 3.4: *Results for Loscos et al. [LDR00]. (a) The original real scene. (b) The relit synthetic scene, with different light sources enables. Courtesy of Loscos et al.*

the BRDF values of the real scene objects as for some methods the focus lies on generating a scene that *looks* realistic. In general these methods do require a detailed geometric model of the real scene. An example of a relit scene using global illumination [LDR00] is given in figure 3.4. The original real scene is shown in (a), the inclusion of a virtual light source is simulated in (b). The lighting effects are created by first removing the effects from the original light sources and then introducing the new lighting effects.

An example of an application domain for this method is architecture. Being able to virtually change the illumination conditions in the real scene makes it possible to see the impact of a new building in a street under different illumination conditions without the need of recording the real environment under all these different conditions. Another application area is crime investigation [HGM00]: a recording of a scene at a certain time can be changed to the illumination at a different time, making it possible to visualize the perception of the criminal at the time of the crime.

C Physically-based illumination

This last category encloses the methods that make an attempt to retrieve the photometric properties of all objects in the scene. These methods estimate BRDF values as correctly as possible based on the physical laws of light and material interaction. The BRDF values can be estimated using an image-based sampling aproach [War92][MPDW03] or can be calculated based on the photometric equilibrium equations [Mar98][SWI97]. The latter method is often referred to by the term inverse illumination, and it is this method that is considered in this chapter. The obtained BRDF information can be used to simulate common illumination or relighting. The accurate BRDF estimation allows a complete and realistic relighting, which takes both reflections and global illumination into account. Patow et al. [PP03] give an in-depth overview of inverse illumination techniques. An example of inverse global illumination [YDMH99] is illustrated in figure 3.5. In this example, a sophisticated BRDF model is used (specular and diffuse) to represent the materials in the scene. Once the material properties are known, any kind of virtual light source can be applied to the scene and its effects simulated.

3.2.3 Assessing the illumination methods

Mixed reality environments are not necessarily designed to lure the user into believing that what he sees is real. For instance VR often aims at trying to create the perception of a real world, without necessarily using convincing real imagery. Some AR systems merely add data displays to real scenes, making no

(a) (b)

Figure 3.5: *Results for Yu et al. [YDMH99]. (a) The original input scene. (b) The result of illuminating the original scene using a different illumination pattern. The specular and diffuse parameters of the real objects are calculated. Courtesy of Yu et al.*

attempt to mix the two seamlessly. However, the focus of this thesis is on problems associated with illumination methods that generate photo-realistic imagery. Therefore this chapter only considers MR scenes that *do* try to convince the users of *believing* that a real world is surrounding them. The amount of realism is used as a measure to assess the quality of the method.

An MR is convincingly real when it is impossible to separate the virtual elements from the real elements in the resulting environment. Five critical success factors have been identified that need to be present in the MR in order for it to appear convincingly real:

- *The positioning of the object within the real scene needs to be realistic.*

  In other words it is essential to obtain a good scene and camera calibration, even if limited to, for instance, the ground plane of the scene. The object registration should not introduce visible errors, such as incorrect occlusion effects.

- *After including the virtual object(s), the resulting scene needs to have consistent lighting effects.*
  The main difficulty in conforming to this requirement is to find the correct appearance of the new lighting effects. For instance, for virtual shadows it is important to simulate correctly their orientation, colour, and shape. Sometimes the lighting effects are approximated, but they can be calculated exactly if the geometry of the scene, the illumination characteristics and the material properties of all objects in the scene are known.

- *The virtual object(s) must look natural.*
  A cartoon-like virtual object is easily detectable and therefore efforts have been made to model objects that look realistic. One successful technique is *image-based modelling*, in which objects are rendered with textures based on real images.

- *The illumination of the virtual object(s) needs to resemble the illumination of the real objects.*
  There are two possible methodologies to achieve this requirement. Either the illumination pattern of the real scene is known, which in turn is used to illuminate the virtual object, or all material properties of all objects in the scene are known or estimated, which allows the entire scene to be consistently relit with a known illumination pattern.

- *If the user can interact with the MR environment, it is clearly important that all update computations occur in real-time.*
  Any delay in the interaction reminds the user of the fact that what is seen is unreal [MW93].

The requirement of a real-time system is one of the most difficult to achieve, especially when no pre-processing time is allowed.

The ultimate objective of the aforementioned methods is defined by the amount of realism perceived by the user. This inherent subjectivity complicates an objective assessment of the various methods. In this section a few quality criteria are listed that are used in section 3.4 to assess the presented methods:

- **Amount of realism:** in some cases it is impossible to evaluate the amount of realism without using a statistical measure. For instance, a test audience can evaluate the method, if the test group is large enough, a statistical value can be derived from the group evaluation. Alternatively, if the inserted virtual object is an exact replica of an existing real object, it is possible to give an exact value of the amount of realism in the produced scene. It suffices to compare the generated scene with a photograph of the real object in the same scene. The difference between the two gives a measure of the level of realism obtained.
- **Input requirements:** it is expected that the more input data is available, the higher the quality of the end result will be. On the other hand, the usability of the system lessens with the complexity of the input data. Possible input data are: the geometry, the light position, the illumination settings and the material properties.
- **Processing time:** the time needed to create the end result is another important characteristic of the method. To offer the user a highly realistic interactive environment, the computations need to be done in real-time. Unfortunately, this is very hard to achieve. If geometric and material properties of a scene need to be known, it is unavoidable that some pre-processing time needs to be incorporated. In general the usability of the proposed methods depends on the amount of pre-processing time needed and the computation speed of the illumination method.
- **Level of automation**: if the method under consideration requires a considerable amount of manual intervention to process the input data, the method is less interesting than one that is automated.
- **Level of interaction**: a method can be judged based on its dynamic character: the ability to change the viewpoint of the camera, or the ability to let the user interact with the environment, for instance, by moving around objects. A higher degree of interaction leads to greater usability.

3.3 Classification of illumination methods for mixed reality

This section provides a classification of the illumination methods for MR. The actual layout of the classification is presented in section 3.3.1. This classification organizes the methods based on the amount of input data required to execute the illumination method. This results in four different groups. For each of these groups the most relevant illumination methods available in the literature are presented in sections 3.3.2, 3.3.3, 3.3.4 and 3.3.5.

3.3.1 Choice of Classification

MR brings together those applications that create a new environment, around or in front of a user, containing both real and virtual elements. Section 3.2.3 formulated the objectives, the difficulties encountered and the assessment criteria of MR systems. One of these criteria, the type of input requirements, regulates the accessibility and accuracy of the methods. This criterium is used to classify the different methods. An overview of the input information needed to calculate a common illumination, relighting

or inverse illumination solution is given in figure 3.6. The input data are the geometric model, the scene radiance and the object to be included into the scene. This is fed into the illumination simulation system which either produces common illumination, relighting or physically-based illumination, to produce the MR.

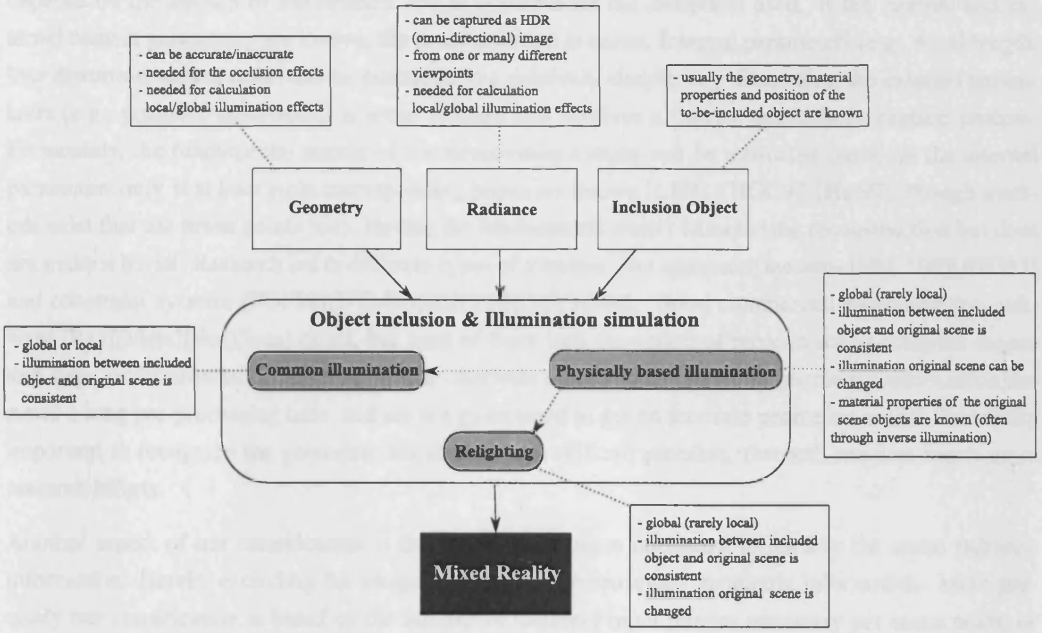


Figure 3.6: *An overview of the dataflow in illumination calculation for MR. Three input sources are considered, indicated by the white square boxes: the scene geometry, the scene radiance and information about the to-be-inserted object(s). These input data are used to calculate the illumination solution. The generation of the illumination between the virtual and real objects provides common illumination, relighting or physically-based illumination. The final output, indicated by the black square, is the MR. The accuracy of the merging depends on the amount of input information available.*

The classification put forward in this chapter firstly takes into account the required geometric model of the real scene, starting with the methods that require no geometric model and ending with methods that require a precise geometric model. In this thesis a *geometric model* is defined as a reconstruction of a part of the (or the entire) real scene with significant detail. A geometric model defines mathematically the surfaces of the scene, either as a set of points or a mesh of triangles. The pre-processing workload for methods that extract a basic geometric model, e.g. the depth at a low resolution or the position of the ground plane, is significantly lower than those methods that do require a high-level geometric model. Therefore methods using basic geometric information are classified under the group of methods that do not require a geometric model, as this gives a better indication of the amount of pre-processing time required for each different class.

Two different approaches exist to reconstruct a geometric model of the real scene. Either the scene is scanned with a scanning device [Nyl][MNP+99][Metb] or it is reconstructed using stereovision [HGC92][Har97][Fau92][Fau93]. The first option of using a scanning device gives a precise geometric model but is expensive and tedious. Often the model captures too much detail, which is not always necessary and is difficult to manage for real-time applications. Objects such as trees and objects with a highly specular surface are for some scanning techniques difficult to model accurately. Instead of using a scanning device, modelling techniques based on stereovision can be used to reconstruct the geometry of a scene. Most methods described here requiring a 3D model of the scene make use of this low-cost

solution. In general, the 3D reconstruction requires at least two photographs captured from two different viewpoints. However, the entire geometry of a scene cannot be captured with one pair of photographs only, this would create gaps in the known geometry. Usually more than one pair of photographs is required for a complete geometric reconstruction. The ease at which this reconstruction can take place depends on the amount of information that is available for the camera(s) used. If the internal and external camera parameters are known, the reconstruction is easier. Internal parameters (e.g., focal length, lens distortion, aspect ratio) can be estimated in a relatively simple way. Recording the external parameters (e.g., position, orientation) is more difficult and involves a precise and tedious capture process. Fortunately, the fundamental matrix of the stereovision system can be estimated based on the internal parameters only, if at least eight corresponding points are known [LH81][HGC92][Har97] (though methods exist that use fewer points too). Having the fundamental matrix can ease the reconstruction but does not make it trivial. Research led to different types of systems: non constraint systems [FRL+98][SWI97] and constraint systems [POF98][DTM96][DBY98][MYTG94]. Good commercial reconstruction software [Rea][Meta][Eos][Inta] exists, but most of them lack the option of reconstructing complex shapes and large environments. In general, we can conclude that systems requiring geometric information demand a long pre-processing time, and are not guaranteed to get an accurate geometric model. It is really important to recognize the geometric acquisition as a difficult problem, that still requires much more research efforts.

Another aspect of our classification is the amount of images necessary to retrieve the scene radiance information. Hereby excluding the image data needed for retrieving geometric information. More precisely our classification is based on the number of different input images necessary per scene point, or per material, to retrieve BRDF information and to render the final scene.

Some projects adopted the concept of HDR imaging [RWPD05], which has been discussed in chapter 2. Each HDRI is generated based on a set of images taken from the same viewpoint of the same scene, but with a different exposure. It may be argued that methods using HDRIs should be classified under that class with methods that use more than one image for each point of the scene. However, in this thesis it is assumed that an HDRI provides one radiance value per point, and methods that use only one HDRI for a certain point of the scene are therefore classified as methods requiring only one input image. Similarly, methods that require a few or many HDRIs are classified as methods using respectively a few or many images.

The following classification is used throughout the remainder of this section:

1. **Model of the real scene unknown, one image known:**
   This category lists those methods that do not require any model of the real scene, except for some low-level geometry like depth information. Any necessary radiance information of a certain point in the real scene is extracted from one single image.

2. **Model of the real scene known, one image known:**
   A geometric model of the real scene is available. Any necessary radiance information is extracted from one image only.

3. **Model of the real scene known, few images known:**
   Again a geometric model of the scene is required. For a certain point in the scene, radiance information is available from a few different images. A *few* in this context means two or three images (e.g, one image to show the scene point in shadow, one to show the scene point not in shadow as presented by Loscos et al. [LFD+99]).

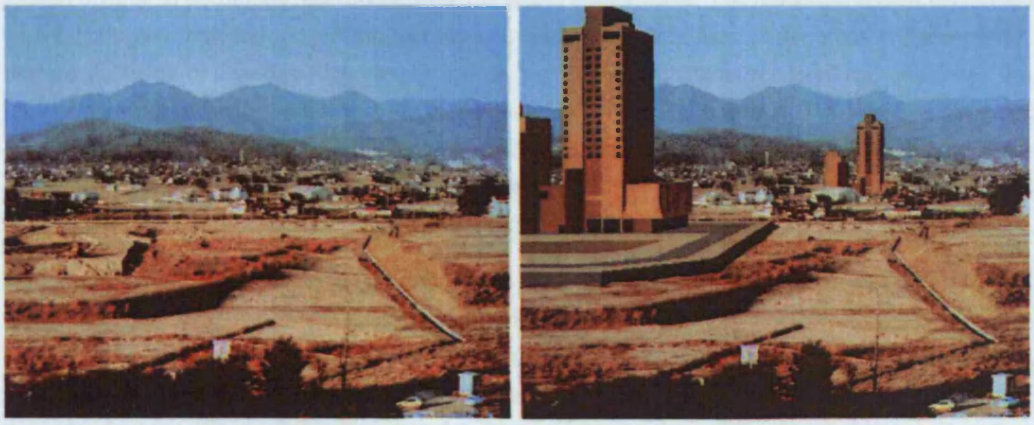
4. **Model of the real scene known, many images known:**
   This class groups those methods that require both a detailed geometric model of the real scene and radiance information from a large set of different images. *Many* in this context means more than three (e.g., a scene point at several different times of the day, or under different illumination settings as presented by Yu et al. [YM98]).

The following sections list the most significant methods based on the above mentioned classification and briefly discusses their methodology. A discussion of the methods based on the assessment criteria mentioned in section 3.2.3 is given in section 3.4.

3.3.2 Model of the real scene unknown, one image known

To this challenging category, in terms of output quality, belong those methods that require very little relevant information about the real scene. Since no geometric model of the scene is available it might be necessary to calculate depth information of the scene, to allow a correct inclusion of the virtual objects, or some illumination information. For this group, all radiance information is extracted from one single image.

Nakamae et al. [NHIN86] were the first to propose a method for composing photographs with virtual elements. Input photographs are calibrated and a very simple geometric model of the real scene is extracted. The viewpoints of the photograph and the virtual scene are aligned to ensure an appropriate registration of the virtual objects within the photographed elements. The sun is positioned within the system according to the time and date when the picture was taken. The sun intensity and an ambient term are estimated from two polygons in the image. The illumination on the virtual elements is estimated and adjusted to satisfy the illumination in the original photograph. The composition is done pixel by pixel and at that stage it is possible to add fog. All parameters are very inaccurate and therefore the results are limited in accuracy. However, they were one of the first to mention the importance of using a radiometric model to improve the image composition. Figure 3.7 displays an example of the obtained results.



(a) (b)

Figure 3.7: *Results for Nakamae et al. [NHIN86]. (a) The original background scene. (b) The augmented scene where the illumination of the augmented objects are matched to their surroundings and the shadows are cast accordingly. Copyright Nakamae et al.*

Methods exist in computer graphics that use *environment maps* to render objects in a scene. They were introduced to approximate reflections for interactive rendering [BN76][Gre86][VF94]. These methods can also be used to assist the rendering of glossy reflections [CON99][HS99][KM00] by pre-filtering a

map with a fixed reflection model or a BRDF. At this moment, graphics cards extensions support the real-time use of environment maps, this encourages its use even more. Graphics cards now support cube maps [NVi]; ATI [ATI02] presented at SIGGRAPH 2003 a demonstration of a real-time application for high resolution. Environment maps can be used to represent the real scene in an MR environment as a panorama and the information from these images can be used to simulate reflections on a vertical object positioned at the centre of the environment map [Che95].

(a) (b)

Figure 3.8: *Results for Agusanto et al. [ALCS03]. (a) The virtual objects are rendered with skin textures. The left object is blended with a diffuse map and no soft shadows. The right objects is blended with a glossy map and with soft shadows. (b) The original lightprobe image, and the glossy environment map. Courtesy of Agusanto et al.*

Agusanto et al. [ALCS03] exploited the idea of environment maps to provide reflections in AR. They use HDRIs of the environment captured by a light probe to create the environment map. These maps are filtered off-line to decompose the diffuse from the glossy components. The rendering is then performed with a multi-pass rendering algorithm that exploits hardware capabilities. After some pre-processing, like the inclusion of shadows, they present results for MR environments rendered on a desktop. An impressive aspect of their work is that the method also works for real-time AR. The implementation of their method works with ARToolKit [KBBM99] and the results show interactive reflections from the real scene on virtual objects at interactive frame rate. An example of such a projection is given in figure 3.8. Although it should be feasible, they have not yet provided a shadow algorithm for the AR application.

Sato et al. [SSI99] adopt a technique that extends the use of environment maps to perform common illumination. In their method, it is assumed that no geometry is known a-priori. However, at least a few images are known from different but very restricted and known viewpoints, which can be used to estimate a very simple geometry of the scene and the position of the light sources. The obtained geometry does not offer a reliable occlusion detection and the positions of the virtual object are therefore restricted to lie in front of all real objects in the real scene. After this low-level geometric reconstruction, a set of omni-directional images are captured with varying shutter speed. From these images, a radiance distribution is calculated, which in turn is mapped onto the geometry. To calculate the shadows and the local illumination a ray casting technique is adopted. The radiance values of the virtual objects are calculated using the information known about the light sources, the radiance values of the real scene, the geometry and the BRDF values of the virtual objects. To simulate the shadows cast by virtual objects onto real objects, the radiance values of those points in the scene that lie in shadow are scaled. The

simulated soft shadows look realistic, see figure 3.3. Their geometric estimate is poor and therefore usability of the method is limited and the positions of the virtual objects are restricted. Nevertheless the method produces convincing local common illumination. The presented method relies on the geometry to identify which points need to be scaled to simulate virtual shadows. Problems are expected when the virtual shadow overlaps with lighting effects already present in the scene radiance. An inconsistent scaling, or misaligned scaling will create artefacts. This is ignored by Sato et al., instead they avoid the situation when overlap between virtual and real lighting effects takes place.

3.3.3 Model of the real scene known, one image known

Most of the existing illumination methods assume that a geometric model of the scene is available. The more detailed the geometric model is, the more reliable the occlusion detection will be. Although not all methods explain where this model should come from, it is doubtful that a perfect geometric model can ever be acquired and this should be taken into account when evaluating a specific method. In this section a discussion is given of those methods that take a certain 3D geometric model of the real scene as input and extract radiance information from one single image. All methods that belong to this category are further divided into three groups based on the type of illumination they produce: local illumination for AR applications, common illumination, or relighting.

A Local illumination for AR

As mentioned before, AR has long been an area wherein people focused on registration and calibration as these are still difficult problems to solve in that area. However, a few papers tried to introduce shadows in their systems, to show how well the registration was done and to improve the rendering quality. Recent improvements in graphics hardware for rendering shadows made it possible to perform real-time rendering of shadows on well-registered systems where the geometry is known. Early work was presented by State et al. [SHC+94] in which virtual objects are inserted in the see-through real scene. A real light source is moved around and tracked, and shadows of the virtual object due to this real light source are virtually cast onto real objects by using the shadow mapping technique described in [Bli88]. In this case, the light source is assumed to be a point light source. It was very promising that some researchers in AR were interested in using local common illumination in their systems, but it was followed by a long period in which no innovative material emerged. Recently, additional work of Haller et al. [HDH03] has been presented which adds shadows from a virtual object onto real objects. The methods uses shadow volumes to create the virtual shadows, and in order to get good quality results knowledge about the scene geometry is essential. Methods using shadow volumes scale the pixel values of the real texture that fall within the shadow volume using one scaling factor. Shadow volumes techniques do not necessarily distinguish between points that are or are not already inside a real shadow, and as such will create incorrect virtual shadows when virtual and real shadows overlap. Haller et al. ignore the artefacts created when such an overlap occurs.

B Common illumination

Jancene et al. [JNP+95] use a different approach to illuminate the virtual objects. They base their method, called RES (Reality Enriched by Synthesis), on the principle of composition. The objective is to add virtual animated objects in a calibrated video sequence. The final video is a composition of the original video sequence with a virtual video sequence that contains both virtual objects and a representation of the real objects. The geometry of the real object is reconstructed a-priori so that for each frame in the video the geometry is known. The rendering in the virtual sequence is performed using ray tracing. It is possible to modify the reflectance properties of real objects. Shadows are simulated

in the virtual sequence, and the impact of these shadows in the final video is acquired by modifying the original video with an attenuation factor. An occlusion mask is created to reflect occlusion between virtual and real objects. This method came quite early in the history of common illumination and video composition, and even though it is not applicable for real-time applications, it allows local common illumination and virtual modification of the reflectance properties of real objects. The images on the left in figure 3.9 illustrate the original scene, the images on the right illustrate the composition. The attenuation factor is derived from the reconstructed geometry, therefore misalignment of the real shadow visible in the texture with that calculated through the geometry might result in artefacts when virtual shadows overlap with those real shadows. Jancene et al. do not present a solution for this problem.

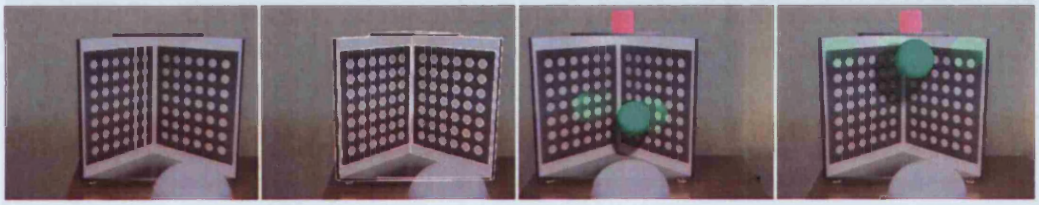


Figure 3.9: *Results for Jancene et al. [JNP+95]. The images on the left hand side show the original scene and the registration of the cardboard box within this scene. The images on the right hand side show two screen shots from the video sequence in which a virtual dynamic green ball and static pink cube have been added to the original scene. The reflection of the green ball is visible on the board behind it. Courtesy of Jancene et al.*

Gibson and Murta [GM00] present a common illumination method, using images taken from one viewpoint that succeeds in producing MR images at interactive rates, by using hardware accelerated rendering techniques. Apart from constructing the geometry of the scene, the pre-processing involves creating a set of radiance maps based on an omni-directional HDRI of the entire scene. New virtual objects are rendered via a spherical mapping algorithm, that maps the combination of these radiance maps onto the virtual object under consideration. To simulate the shadows, a set of M (distant) light sources are identified, which imitate the true, unknown illumination in the scene. Each light source is assigned a position and two parameters α_i and I_i, which define the colour of the shadow. For each light source, a shadow map is calculated. Shadow mapping is an intensive technique supported by the graphics hardware that helps to create shadows in a fast and efficient way. The shadows created with shadow maps are in nature hard shadows and therefore unsuitable for realistic shadow generation. Gibson and Murta combine the M shadow maps in a specific way, using the above-mentioned parameters and succeeds in simulating soft shadows, looking almost identical to the solutions obtained with a more computational and traditional ray-casting algorithm, see figure 3.10. The system of M light sources needs to be defined so that it represents a close replica to the current illumination system, an increase in number of light sources affects the rendering time. To demonstrate their method, Gibson and Murta used eight light sources to simulate an indoor environment. The position and the parameters of the light sources are defined via an optimisation algorithm, which needs to be executed only once for each different scene. For each light source a shadow map is generated using a two-step algorithm. In the first step all shadows are generated, including those of the real objects, in the second step the shadows from the real objects are removed from the shadow map generated in the first step. The result is an augmented scene with global common illumination including soft shadows, however, the addition of inter-reflections is not taken into account. The method does not discuss the problem associated with inaccurate geometry but it is expected that if the geometry is inaccurately known that the removal of the real shadows from the generated shadow map will be incorrect and will introduce artefacts when a pixel inside a real shadow is scaled again to simulate a virtual shadow, due to incorrectly assuming that pixel does not belong to a real shadow.

(a) (b)

Figure 3.10: *Results for Gibson et al. [GM00]. Comparison of a ray-traced (a) and a hardware generated image (b). The ray-traced image was generated using* RADIANCE *[War94], the hardware generated image made use of the rendering method described in [GM00]. The generation of the ray-traced image took approximately 2 hours, the generation of the hardware rendered image took place at nearly 10 frames-per-second. Courtesy of Gibson et al.*

Debevec [Deb98] presents a more advanced common illumination method that estimates the BRDF values for a small part of the scene. It is argued that if a virtual object is inserted into the scene, only a small fraction of the scene experiences an influence from that inclusion. Relighting methods using inverse illumination therefore only require the BRDF values of the points that lie in this fraction. Since for most applications it is possible to know the position of the virtual objects, Debevec uses this position to divide the entire scene into two parts: the *local scene* and the *distant scene*. The local scene is that fraction of the scene whose appearance might alter after inclusion and the BRDF of the materials in that part need to be estimated. The distant scene is that part of the scene that undergoes no physical alteration after inclusion. A schematic overview of the division in local and distant scene and their corresponding influences is presented in figure 3.11 (a). The local scene is restricted to be diffuse only; the distant scene has no restrictions. An omni-directional HDRI is captured using a reflective sphere. The resulting *lightprobe image* is used to present the illumination in the real scene. Based on the geometric model, the light probe image and the division into local and distant scene, the BRDF values in the local scene are estimated. The calculations are straightforward, since only diffuse BRDF values are considered. A *differential rendering* method was developed to reduce the possible inconsistencies in the geometric model and the (specular) error on the BRDF estimates to an acceptable level. The rendering is a two pass mechanism. First, the augmented scene is rendered using a global illumination technique, the result is denoted by LS_{obj}. Next the scene is rendered using the same global illumination method, without including the virtual objects, denoted by LS_{noobj}. If the input scene is represented by LS_b, then the difference between LS_b and LS_{noobj} is exactly the error that results from an incorrect BRDF estimation. The differential rendering therefore calculates the final output rendering LS_{final} as:

$$LS_{final} = LS_b + (LS_{obj} - LS_{noobj})$$

This differential rendering method removes most of the inaccuracies and in a certain way it is similar to the one of Jancene et al. [JNP+95] presented above. The results of this method are promising, see figure 3.11, but it still suffers from a few deficiencies. Firstly, only diffuse parameters of the local scene are estimated, this introduces an error that should be compensated by the differential rendering. Secondly, the viewpoint can be altered but the method is too slow to work at interactive rates. If the rendering could be accelerated using low-cost graphics hardware, it could be possible to achieve interactive update

rates for the MR. The lightprobe image is used as if the distant scene lies at infinity and that it has been captured with a camera infinitely far from the lightprobe. This will introduce a distortion in the alignment between radiance values retrieved from the lightprobe and the scene geometry. Therefore it is expected that the BRDF calculations will be erroneous when the distortion affects an area containing the light sources.

Figure 3.11: *Debevec et al. [Deb98]. (a) The real scene is divided into a local scene and a distant scene. The illumination from the distant scene influences the local scene and the virtual objects. The virtual objects influence the local scene. The local scene and the virtual objects do not have an influence on the distant scene. (b) The original real scene. (c) Using differential rendering and the estimation of the diffuse BRDF estimates, the virtual lighting effects are simulated. Courtesy of Debevec et al.*

Gibson et al. [GCHH03] developed a method to create soft shadows using a set of shadow maps. They created a rapid shadow generation algorithm to calculate and visualize the shadows in a scene after the material properties of the scene are calculated. A proper estimate of both the geometry and the radiance information in the real scene needs to be available. It is assumed that the BRDF for all materials is diffuse. This diffuse BRDF is estimated using geometry and radiance information (one radiance image per 3D point). In their method, the scene is divided into two parts: one part contains all patches in a scene that are visible from the camera, called the *receiver patches* and another part contains those patches in the scene that have a significant radiance, called the *source patches*. These patches are used to build a shaft hierarchy between the receiver patches and the source patches. The shaft hierarchy contains information on which patches block receiver patches from other source patches. Next the scene is rendered from a certain viewpoint. This rendering is a two-pass mechanism. A first pass, goes through the shaft hierarchy to see which source patches partially or completely illuminate a receiver patch. Once these source patches are identified, the radiance of each receiver patch is set to the sum of all irradiance coming from these source patches, without taking occlusions into account. The second rendering pass, takes the shadows in consideration. To calculate the portion of blocked light, a shadow mapping technique is used. In fact, a shadow map is created for each source patch. At each receiver patch, these maps are then combined and subtracted from the radiance value that was rendered in the first pass. This method is capable of producing soft shadows in a fast and efficient way. In figure 3.12 examples are given of synthetic scenes rendered using the above described method. Renderings of the same synthetic scenes using a ray tracing method and photographic reference images are given as well. The shaft-hierarchy is build from the geometric model available, therefore it is likely inaccuracies will be present when the geometric model is incorrect.

Another set of methods were built to exploit the structure of a radiosity method. Fournier et al. made pioneering work in this direction [FGR93]. When this method was developed, facilities for modelling a geometric model from a real scene were not available. To overcome this issue, Fournier et al. decided to replace the geometry of the objects in the real scene by their bounding box, and an image of the object was applied on each of the faces of the box. An example of such a model is shown in figure 3.13. To

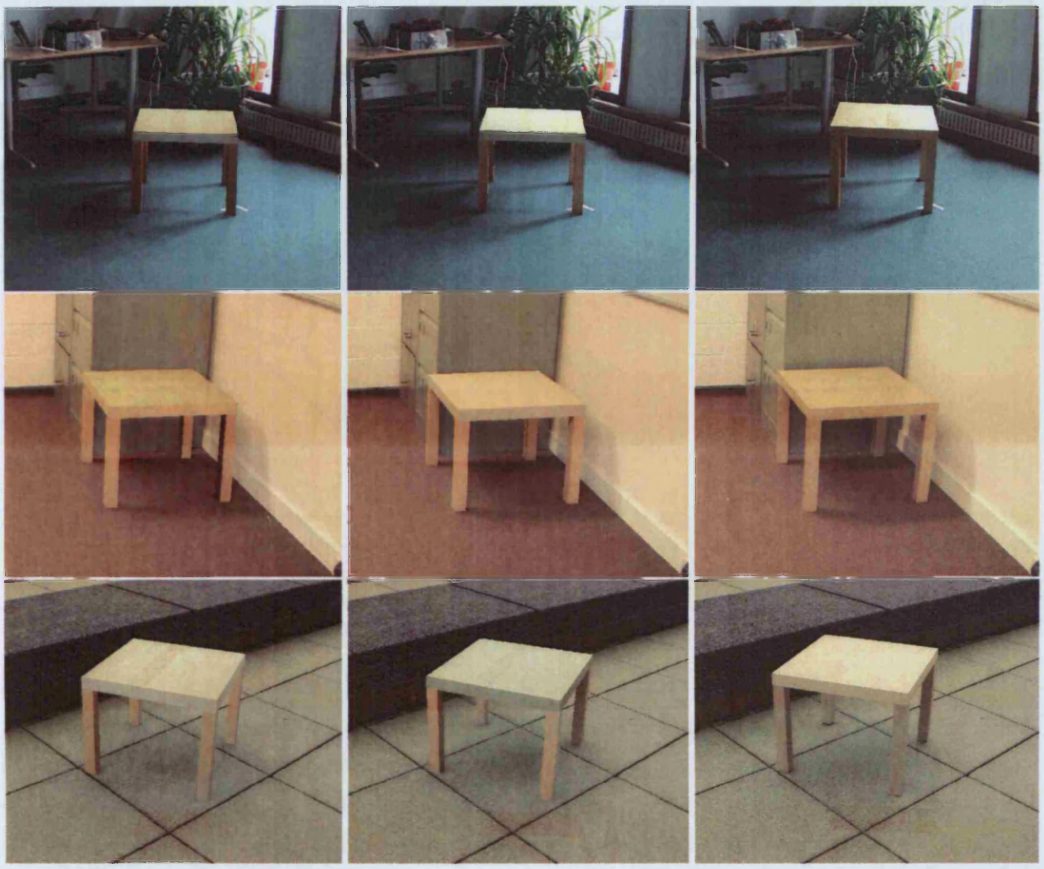
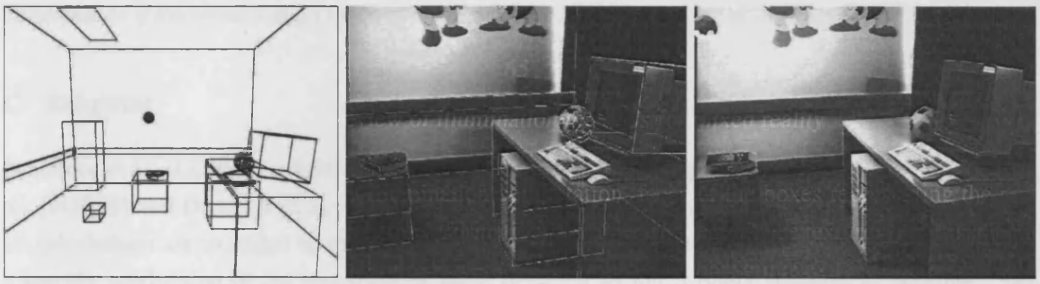


Figure 3.12: *Results for Gibson et al. [GCHH03]. A comparison of the rendering quality for three different scenes. The images in the left column are produced using the system presented in [GCHH03]. The images in the middle column are rendered using ray tracing. The images in the right column are photographic reference images. Courtesy of Gibson et al.*

set up the scene for global common illumination computation, faces of the boxes representing the real objects are divided into patches. Using the information contained in the radiance textures, a diffuse local reflectance is computed by averaging pixels covered by each patch. Light source exitances are estimated and the radiosity of the patches are set as an average of the per-pixel radiance covered by each patch. After insertion of the virtual objects and the virtual light sources in the model of the real scene, new radiosity values are computed for the elements in the scene using *progressive radiosity* [CCWG88]. The rendering is carried out by modifying the intensity of each patch with the ratio obtained by dividing the new radiosity by the original one. In figure 3.13 an illustration of the result of this method is given. The results of this method look promising but it suffers from the lack of a detailed geometry. This leads to misaligned shadows and other types of mismatching between real and virtual objects. The method is slow and does not allow real-time interaction. Nevertheless, this pioneering method has influenced subsequent research work, e.g. Drettakis et al. [DRB97] and Loscos et al. [LDR00] as presented in the remainder of this section.



(a) (b) (b)

Figure 3.13: *Results for Fournier et al. [FGR93]. (a) All objects in the scene are represented by a box, that narrowly fits the object. (b) Image information is mapped on the boxes (note that for the ball, a more complex shape was used). (c) A synthetic rendering showing a virtual object (book positioned on other book) and a virtual light source. The global common illumination effects are generated with an adaptive progressive radiosity algorithm. Copyright Canadian Information Processing Society.*

Drettakis et al. [DRB97] present a method that builds on Fournier et al. [FGR93], but uses a finer model of the real scene. The same equations are used to estimate the light sources emittance, the reflectance of the patches and the original radiosity. Drettakis et al. make use of the more recent hierarchical radiosity method hierarchical [HSA91] accelerated by using clustering [RPV93][Sil95][SAG94]. Based on [DS97] a hierarchy of shafts is built from the real scene model, which allows a local understanding when virtual objects are added. This permits an easy identification of all patches that need to undergo a radiosity alteration due to the influence of the newly added object. The advantage of this shaft hierarchy is that it permits interactive updates of the illumination in the augmented scene when virtual objects move. The final display is made similarly to the method of Fournier et al. [FGR93]: the intensity of the patches is modified with the ratio defined by the modified radiosity divided by the original radiosity. This type of rendering is fast, compared to a ray tracing method, as it uses the hardware capability to render textured polygons. This method provides global common illumination with possible interaction. Unfortunately, the method does not allow changing either the current illumination or the current viewpoint. Also, the shaft hierarchy is built from the geometric model of the scene, therefore if the geometric model is inaccurate, the simulated virtual lighting effects might introduce artefacts when overlapping with real lighting effects, due to the misaligned or incorrect projected scene radiance onto the geometry. In figure 3.14 a screen shot is given of the 3D reconstruction and an example of the resulting MR using the above explained method.

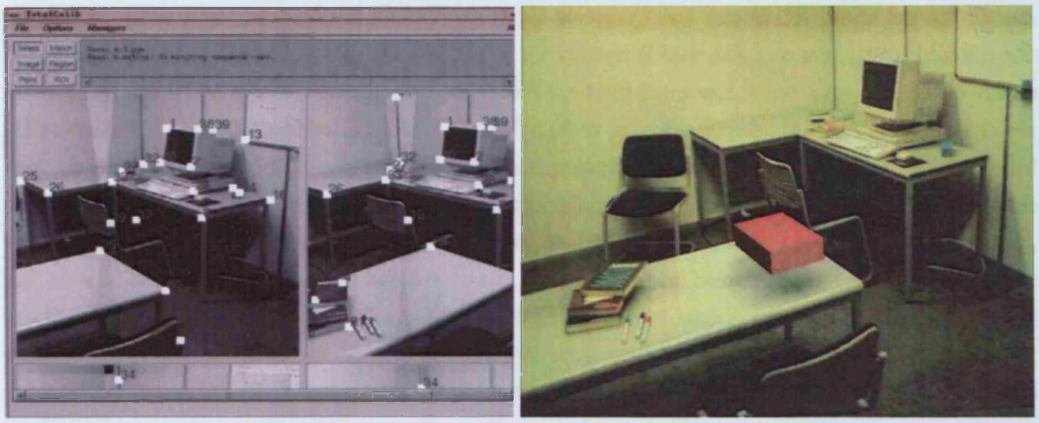


Figure 3.14: Results for Drettakis et al. [DRB97]. In the left image a screen shot is given of the 3D reconstruction of the real scene. The right image gives an example of the MR, the floating box is the virtual object. The virtual objects can be moved at interactive rate while keeping the global illumination effects. This is carried out by using an adaptation of hierarchical shafts for hierarchical radiosity [DS97]. Courtesy of Drettakis et al.

C Relighting

In Loscos et al. [LDR00], relighting is made possible, while keeping the framework set by Fournier et al. [FGR93] and Drettakis et al. [DRB97]. The scene parameters are extracted in the same way but all calculations are extended to the use of HDRIs [Los99]. Since this method focuses on relighting, a specific subdivision of the real scene is made to detect as much direct shadows as possible. The radiosity of each element is modified to simulate non-blocked radiosity, in other words, to erase the shadows from the textures. A factor is computed using the radiosity method without taking the visibility in consideration. Then the new radiosity value is used to update the texture. Approximations of the estimation and of the input data lead to an inexact modification of the texture. In a second step, another factor is applied to automatically correct the imprecisions. This is done by using a reference patch that reflects the desired result. Once this is done, the new textures are used instead of the original ones, and reflectance and original radiosity values are updated accordingly. Shadows can be simulated using the factor of the newly computed radiosity solution divided by the original radiosity (without shadows). This method also extends the method presented in [DS97] for the insertion of virtual lights. In the system of Loscos et al. [LDR00], it is possible to virtually modify the intensity of real light sources, to insert virtual objects that can be dynamically moved and to insert virtual light sources. The problem that comes with inserting new lights or increasing light source intensity is that the value of the factor computed between the new radiosity value, divided by the original radiosity, may be greater than one. In that case, multi-pass rendering is used to enable the visualisation of brighter illumination. This method allows interactivity and is fairly rapid in the pre-processing computation. However, the obtained results are inaccurate as the illumination of the real scene is not fully estimated. Firstly, because lit areas are not altered at all, and secondly, because it concentrates on the diffuse component only. An example of the results was already shown in figure 3.4.

Although it does not seem feasible to estimate specular components of the BRDF from one single image, Boivin et al. [BG01] present a method that re-renders diffuse and specular effects based on radiance information from one single image and a full geometric model of the scene, including the light source positioning and the camera properties. With a hierarchical and iterative technique they estimate the reflectance parameters in the scene. In this method, the reflectance model of Ward [War92] is used, which presents the entire BRDF with either 3 (isotropic materials) or 5 (anisotropic materials) different

parameters. The BRDF estimation process starts by assuming that the BRDF values are all diffuse. A synthetic scene is rendered using the geometry, the current BRDF estimate and global illumination methods. If the difference between the real scene and the synthetic scene is too large, the BRDF values are re-estimated using a more complex BRDF model. First specular effects are added and a roughness factor is estimated using a time-consuming optimisation process. Later anisotropic effects are introduced and the optimisation continues until a reasonable synthetic scene is acquired. This is very similar to the way parameters are estimated in [YDMH99]. However, in this case, only one input image is used, and anisotropic parameters are estimated as well. The method of Boivin et al. relies on one single image to capture all photometric information. The advantage of such an approach is that the image capturing is relatively easy; the disadvantage is that only partial geometric information is available: there is no information for those surfaces that are not visible in the image. Nevertheless,the proposed method allows changing the viewpoint. If a sufficiently large portion of a certain object is visible in the image, the reflectance properties of the missing parts of the object are calculated based on this portion. Grouping objects with similar reflectance properties makes this process more robust. On the other hand, this requires that not only the geometry needs to be known, but also a partitioning of the scene into objects with similar reflectance properties, which may compromise the applicability of this method. Although optimised, the rendering algorithm is computationally expensive and therefore only a non real-time solution can be obtained. The results presented in [BG01] show a scene containing manually modelled light sources with a known position and intensity. The method does not address the applicability of the method on scenes containing un-modelled light sources. In figure 3.15 an illustration is given of the output results of the described method.

3.3.4 Model of the real scene known, few images known

If more information about the radiance of the points in the scene is available, a better BRDF estimate can be acquired. The radiance perceived at a certain point depends on the viewing angle, on the angle of incident light and the BRDF. Hence, it is possible to gain more information about the BRDF of a certain point in the scene if radiance information is available from images captured from a different viewing angle. Alternatively, if the viewpoint is kept the same but the position of the light sources is changed, extra BRDF information is captured as well. In this section, the methods are discussed that make use of this extra information.

Loscos et al. [LFD+99] developed a system that allows relighting, as well as virtual light source insertion, dynamic virtual objects inclusion and real object removal. They identified that it is difficult to estimate reflectance values in shadow regions due to saturation and because this estimate depends on the quality of the indirect light estimation. This is compensated for by adding extra photographs captured under different illumination. The geometry of the real scene is modelled from photographs. This geometric model is textured using one of the images, taken from the different viewpoints. A set of pictures is then taken from this chosen viewpoint while a light source is moved around the scene to modify the illumination. These pictures can be HDRIs as used in [LD00]. The position and intensity of the light sources is known in advance, which limits the applicability of the method. Loscos et al. decided to mix a ray-casting approach to compute the local illumination and a radiosity approach to compute the indirect illumination. Two sets of reflectance values are thus computed. First diffuse reflectance values are computed for each pixel of the viewing window. This is done with a weighted average of the reflectance evaluated with each input image differently lit. The applied weight is based on whether the 3D point associated with the pixel is in shadow relative to the light source position, and also whether the radiance value captured is saturated. The reflectance values are then used to initialize a radiosity system similar to those in [DRB97][LDR00]. This reflectance is then refined by an iterative algorithm. With

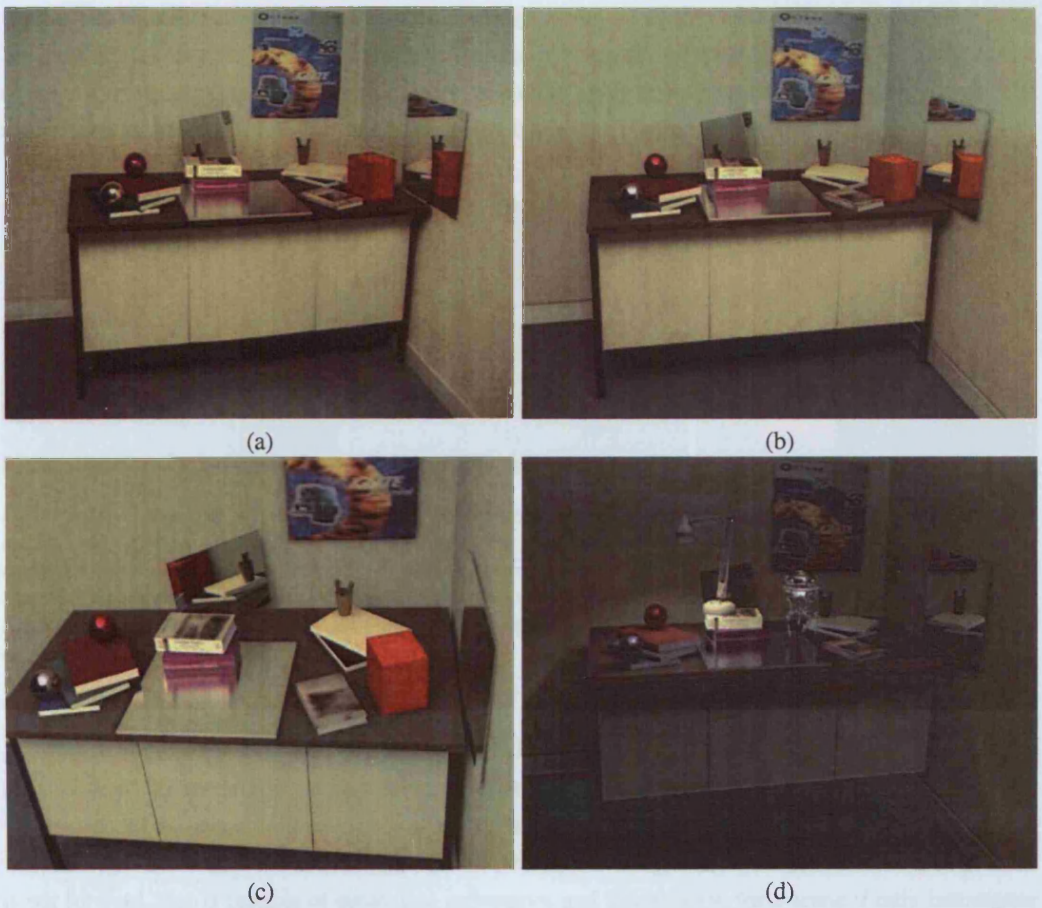


Figure 3.15: *Results for Boivin et al. [BG01]. (a) The original real scene. (b) A relit synthetic image. Diffuse and specular effects are simulated using an optimization algorithm. (c) The viewpoint can be altered, missing information is effectively retrieved through scene segmentation. (d) The scene is relit using a novel illumination setting. Courtesy of Boivin et al.*

this reflectance, Loscos et al. are able to relight the scene using global illumination. Pixel values are updated by adding the local illumination value, computed by ray casting, to the indirect illumination value, computed by hierarchical radiosity using a rough subdivision of the scene. Local modifications are made after the insertion or moving of virtual objects by selecting the area of the window where local illumination will be affected. Indirect illumination is modified by adapting the method of [DS97]. Similarly, virtual light sources can be added, and intensity of real light sources can be modified. A very interesting application of this method is the removal of real objects. The unknown information previously masked by the object is filled using automatic texture synthesis of a sample of the image of the reflectance values of the previously hidden object. The results show that the relighting and the interaction with virtual objects can be achieved in an interactive time. Image examples of the results are shown in figure 3.16. The produced results are good but could be improved by considering specular effects. Due to the nature of the image capture process, it would be very difficult to apply this method on real outdoor scenes. The requirement of needing more than one input image makes the method sensitive to object movement.

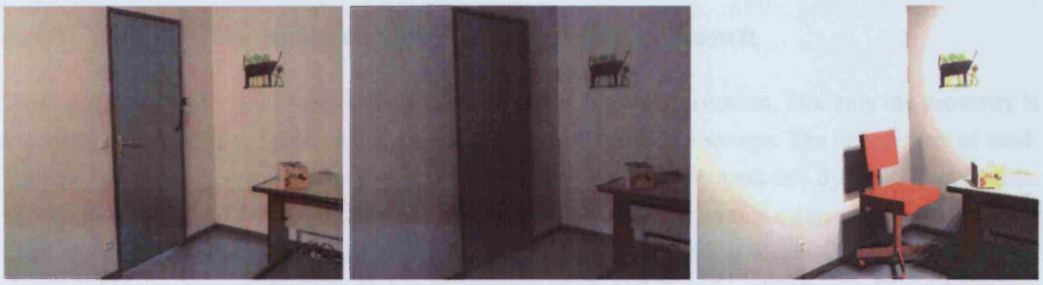


Figure 3.16: *Results for Loscos et al. [LFD+99]. The left image is one of the input images of the real scene. The middle image is a relit image of the real scene, using the calculated BRDF values. The right image illustrates the removal of an object (the door), the insertion of a new virtual object (the chair) and the insertion of a virtual light source. All manipulations are carried out at interactive update rates. The illumination is updated locally with ray casting. The consistency of the indirect illumination is kept using an adaptation of [DS97]. Courtesy of Loscos et al.*

A different approach taken by Gibson et al. [GHH01] results in another relighting method, in which the reflectance of the material is roughly estimated based on a restricted amount of geometry and radiance information of the scene. In theory, only geometry and radiance information is needed for those parts of the scene that are visible in the final relit MR. In their approach a photometric reconstruction algorithm is put forward, that is capable of estimating reflectance and illumination for a scene if only incomplete information is available. To achieve this the direct illumination is modelled as coming from unknown light sources using virtual light sources, see figure 3.17 (a). The aim is not to produce an accurate illumination model, but rather a model that produces a similar illumination as in the original scene. The model used is a spherical illumination surface: a set of small area light sources that surrounds the known geometry. The parameters of this surface, the position and emission of the light sources, are estimated using an iterative minimization algorithm. The reflectance of the materials in the scene are estimated as well. The MR scene is rendered using a ray tracing algorithm. User interaction is impossible at real-time update rate but nevertheless the method illustrates the possibility of getting fairly realistic mixed realities, without limiting input requirements. This method is original, interesting and very practical to adapt to many situations where information on a real scene is partially known. The estimation of the illumination surface uses several input images of the scene, which might limit the applicability of the method to scenes containing dynamic illumination and dynamic or non-rigid objects. An example of a rendered scene and its comparable real scene are given in figure 3.17 (b,c).

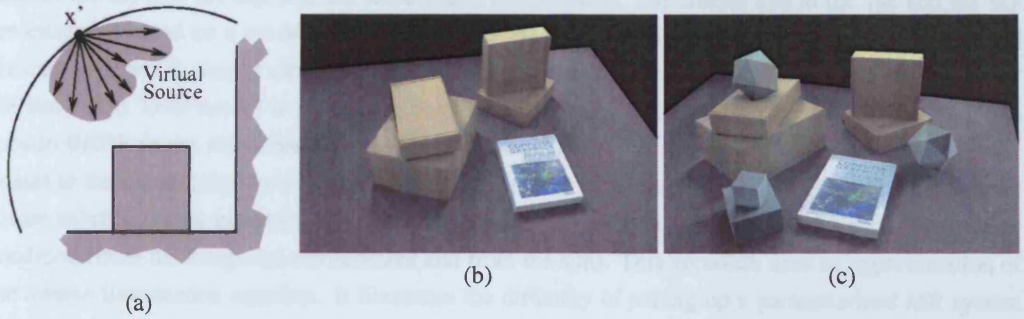


Figure 3.17: *Results for Gibson et al. [GHH01]. (a) The real illumination is approximated by an illumination surface. This illumination surface is covered by a set of virtual light sources. The parameters of these virtual light sources are estimated such that its effect resembles the real illumination. (b) The original real scene. (c) A relit scene containing virtual objects and light sources. Both specular and diffuse effects are simulated. Courtesy of Gibson et al.*

3.3.5 Model of the real scene known, many images known

This category collects those methods that require the most input information. Not only the geometry is required but also radiance information under many different geometric set-ups. The requirement of needing many input images, makes these methods sensitive to object movement and dynamic illumination, especially. Two significant methods could be identified that belong to this category of MR methods. They were selected from a broad set of methods on inverse illumination because they provide a solution for a large group of objects, which is essential for MR considered in this thesis. The first inverse illumination method [YDMH99] focuses on the BRDF estimation, using many HDRIs from different viewpoints. The second [YM98] allows to relight outdoor scenes. This section briefly discusses these two methods.

Yu et al. [YDMH99] use a low parametric reflectance model, which allows the diffuse reflectance to vary arbitrarily across the surface while non-diffuse characteristics remain constant across a certain region. The input to their system is the geometry of the scene, a set of HDRIs and the position of the direct light sources. An inverse radiosity method is applied to recover the diffuse albedo. The other two parameters in the reflectance model of Ward [War92], the roughness and the specular component, are estimated by a non-linear optimization. For the estimation of the specular BRDF, it is assumed that many HDRIs are available from a different set of viewpoints. The estimation makes use of the position of the light sources and the possible highlights they may produce on a surface due to specular effects. It is therefore helpful to capture images of the scene with various numbers of light sources, since this might increase the number of specular highlights. This precise estimate of the BRDF values in the scene allows to remove all illumination in the scene and a new illumination pattern can be applied. To render the scene they make use of Ward's RADIANCE system [War94]. No further steps were taken to speed up the rendering process. The results were already presented in figure 3.5 which illustrates the original real scene and a relit scene using a novel illumination setting. This method is interesting for MR because it provides an algorithm to estimate an accurate complex BRDF of a complex real scene, resulting in an accurate representation of the illumination. The method uses modelled light sources, and therefore it is less applicable for real-life environments with uncontrollable light sources.

Yu and Malik [YM98] present a method that allows relighting for outdoor scenes based on inverse illumination. As it is impossible to retrieve the geometry of the entire scene, they separate the scene into four parts: the local model, the sun, the sky and the surrounding environment. The illumination

sources are the sun, the sky, and the surrounding environment. Luminance due to the sun and the sky are estimated based on a set of input images. At least two photographs per surface of the local model are captured, which should show two different illumination conditions (directly and not directly lit by the sun). The local model is subdivided into small surfaces. Based on these two photographs, two pseudo-BRDF values are estimated per surface. One relates to the illumination from the sun, the other relates to the illumination from the integrated environment (sky plus surrounding environment). A least square solution is then used to approximate the *specular term* for each surface and for each illumination condition (from the integrated environment and from the sun). This approach uses an approximation of the inverse illumination equation. It illustrates the difficulty of setting up a parameterized MR system for outdoor scenes. At rendering time, different positions of the sun are simulated. After extracting the sun and the local model from the background, sky regions are identified and they are mapped on a mesh supported by a hemisphere. Three parameters control the sky intensity. A first scale factor is applied when simulating sunrise and sunset; it is constant otherwise. The second parameter adjusts the intensity of the sky depending on the position of the mesh on the dome. A last parameter controls the sky intensity depending on the sun's position. Next, the radiance values and the pseudo-BRDFs are used to reproduce the global illumination on the local scene. This method is the first to present the possibility of relighting outdoor scenes. Results of these relit scenes and a comparison image are shown in figure 3.18. Although it is difficult to evaluate the quality of the relighting from the images provided by the authors, the images resemble the real conditions, and this can satisfy most of the MR applications for outdoor environments. The drawback of the method is the requirement to have images of the scene under various different illumination settings. The outdoor scene shown in their example is lit by the sun and a blue sky, no clouds are visible. Such a lengthy capture however, will in several environments be difficult to control.

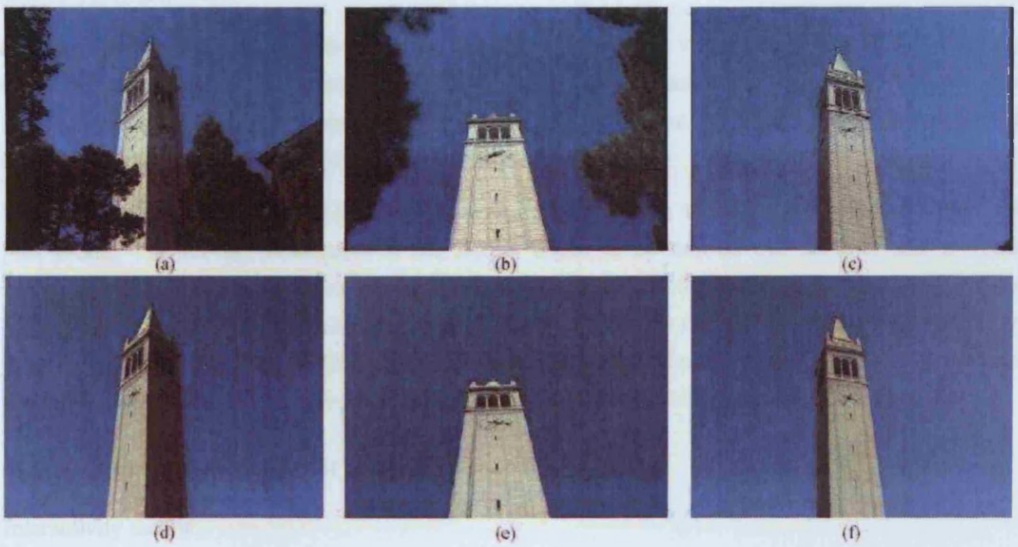


Figure 3.18: *Results for Yu et al. [YM98]. The top row images illustrates the original tower from different viewpoints. The bottom row are synthetic images of the tower from approximately the same viewpoint. The real images were not used to generate the synthetic images, nevertheless the synthetic and real images look very similar. Courtesy of Yu and Malik.*

3.4 Discussion of illumination methods

In section 3.2.3 we pointed out that the assessment of the various illumination methods for MR comes with a certain degree of subjectivity. Fortunately there are some aspects that can be evaluated in a rather objective way. Some of these measures are used in this section to assess the previously listed illumination methods. Section 3.4.1 discusses the amount of *pre-processing* required to simulate the lighting effects in the scene. In section 3.4.2 an evaluation of the *degree of interactivity* allowed by the illumination methods is given. Section 3.4.3 discusses how the *quality* of the simulated illumination can be assessed and uses this to evaluate the methods classified. An overview of the discussed methods is given in section 3.4.4.

3.4.1 Pre-processing

The term *pre-processing* refers to those steps, carried out once, that are required by the method before the merging of real and virtual objects takes place. The geometric reconstruction, image capturing, and BRDF estimation are considered as pre-processing steps.

The methods of the first group, described in section 3.3.2, do not require a full geometric model of the real scene, examples are Sato et al. [SSI99] and Nakamae et al. [NHIN86]. For these methods information about the ground plane can be sufficient to generate the lighting effects. Nevertheless this usually implies a very limited inclusion, such as only positioning of the virtual object onto the foreground or ground plane. All other illumination methods require a geometric model. Some of these methods do not explain how this model can be constructed, others assume that it is constructed using user-aided 3D reconstruction software. Using reconstruction software usually results in a low resolution model and is in general error-prone. This is due to the fact that no automatic, accurate 3D reconstruction software is yet commercially available. Scanning devices give a better resolution, but these devices are expensive and while the scanning of a small object might be straightforward, the scanning of a larger scene is laborious. As a summary we can say that a perfect geometric model is difficult to acquire and that reconstruction is always a tiresome work.

Some methods require radiance information captured from several viewpoints [YDMH99][GHH01] or under different types of illumination [LFD+99][YM98]. Taking several HDRIs from different viewpoints and under different illumination delays the image capture time and makes the method more prone to errors when applied to an uncontrolllable environment.

Many methods calculate a BRDF estimate, some use a diffuse model, some allow a more complex model. Often the calculation of the BRDF needs to be carried out off-line, due to timing issues and is therefore considered as pre-processing work. Methods that calculate a diffuse-only BRDF are: [Deb98][FGR93][DRB97][LDR00][LFD+99], methods that allow specular components are: [GHH01][YDMH99][YM98][BG01]. Usually the estimation of the specular and anisotropic material properties is computationally more demanding than the estimation of the diffuse parameters.

3.4.2 Level of interactivity

Interactivity means:

- the possibility of navigating objects or viewpoints in the scene,
- the effort made to get an interactive rendering,
- the possibility to modify reflectance properties of real objects in the scene,

- the possibility to modify the illumination sources in the real scene.

A few methods allow navigating the virtual objects or the viewpoints. These methods have either enough BRDF information [BG01][YDMH99][FGR93], enough geometry and illumination information [SSI99][YM98] or use a different approach [ALCS03][SHC+94][JNP+95].

Only a few of the methods operate in true real-time (RT) [ALCS03][SHC+94][GCHH03], others are near real-time (near RT) [LDR00][LFD+99][DRB97] but most of them are non real-time (NRT). However, it should be noted that some methods were developed years ago, when computer hardware and software were much slower than nowadays. Also, it should be pointed out that some methods did not specifically attempt to produce interactive systems. With a few modifications, it should be possible to speed up most of the described systems.

Some methods that specifically tried to speed up the computations are worth mentioning. Agusanto et al. [ALCS03] exploited the idea of environment mapping while State et al. [SHC+94] used shadow mapping and Haller et al. [HDH03] shadow volumes. Gibson et al. [GM00] developed a new method to simulate soft shadows at interactive rates and Drettakis et al. [DRB97], Loscos et al. [LDR00] and Loscos et al. [LFD+99] made use of a hierarchical radiosity algorithm, that decreased the computation time to interactive rates as well. Gibson et al. [GCHH03] used shadow maps.

Once the BRDF of all objects in the scene is known, the existing lighting effects can be cancelled. This allows us to change the BRDF of a certain real object in the scene. This can be used to modify the appearances of real objects in the scene. Relighting methods can use this BRDF information to relight a scene using a different illumination pattern. Table 3.1 gives an overview of the various different types of illumination the discussed methods allow.

3.4.3 Quality obtained

Some of the described methods evaluated the quality of their method using one or more of the following evaluation methods:

- A comparison is made between a photographic reference of the real scene and a synthetic version of the same scene.
- The BRDF is measured using an external device and these results are compared with the estimated BRDF values.

Gibson et al. [GCHH03] compare their shadow rendering method with a ray traced rendering and an image of the real scene, see figure 3.12. They are capable of producing realistic and similar shadows as in the real image and at a faster time than the ray traced rendering. In [GM00] the presented extended shadow mapping is compared with a ray traced version using the same input parameters, see figure 3.10. There are some differences between the two synthetic scenes, but the generated shadows look realistic. Boivin et al. [BG01] extract a full BRDF model and compare their rendering with an original image of the real scene, see figure 3.15. In [YM98] the diffuse and specular components are calculated; the resulting rendering is compared with an original image of the real scene. Likewise, [LDR00][LFD+99] estimate the diffuse BRDF and compare a synthetic rendering with an original image of the real scene (see figures 3.4 and 3.16). In both methods, the rendering occurs at interactive update rates. Similarly, Gibson et al. [GHH01], see figure 3.17, compare an original and synthetic image and find that the error between the two images decreases drastically in the first three iterations. Both diffuse and specular reflectance values are modelled. Yu et al. [YDMH99] estimate diffuse and specular BRDF values and

compare these with measured BRDF values of objects in the scene. The estimates and the true values are similar.

We can also compare methods that use both specular and diffuse BRDF values for the rendering with those that have a more restrictive understanding of the BRDF. It is understood that systems based on a more complete BRDF model result in an MR of a higher quality than those based on diffuse BRDF values only or those that do not estimate BRDF values at all. For some methods, only a subjective user perceptive assessment can be made.

3.4.4 Overview

Table 3.1 gives an overview of all methods discussed in section 3.3. For each method, the overview discusses the following aspects:

- Geometric model of the scene: whether or not the method requires a geometric model of the scene.
- Number of different images: the number of different images needed per point in the scene, to calculate the MR.
- Methodology: the methodology used to create the MR. In section 3.2.2 three different approaches were discussed: common illumination, relighting, inverse illumination. Further to this division, a distinction is made between local and global illumination methods.
- Rendering: the rendering method used to compose the MR. Possible answers are: ray-casting, ray-tracing, radiosity, etc.
- Computation time: the *update* time of the method is real-time (RT), non real-time (NRT) or near real-time (near RT).

3.5 This thesis within the classification

The methods discussed in this chapter usually showed results of easy-to-model scenes, with controllable objects and light sources. Only a few methods explicitly tested their method on outdoor scenes [Deb98][GCHH03][SSI99][YM98], nevertheless the outdoor scenes used were fairly restrictive and were not disturbed by object movement or dynamic illumination. Nearly all methods available in the literature assume an accurate geometric model of the scene is available and do not discuss the errors that might be introduced when applying the method on less accurate geometric models. This motivated the research carried out in this thesis.

The focus of the thesis is on illumination methods in general, and therefore methods to improve common illumination and relighting are developed. To be more precise, the common illumination method presented in chapter 4 is based on the method developed by Haller et al. [HDH03]. Nevertheless the framework developed can be incorporated in methods such as provided by Agusanto et al. [ALCS03], Sato et al. [SSI99], [DRB97], and others. The required input data is a poorly reconstructed geometric model of the local scene and one input image for each local scene point. No BRDF values need to be estimated as the shadows are generated using a scaling factor. Thus, the required pre-processing time is rather low. The shadow generation runs in real-time and the entire method is fairly interactive as virtual camera movement and virtual object movement are allowed.

The relighting method presented in chapter 6 presents a radiance capture method which can complement methods such as presented by Debevec et al. [Deb98], Boivin et al. [BG01], Gibson et al.

	3D model	♯ Images	Methodology	Rendering	Computation time	Section
[ALCS03]	/	one	global common illumination	environment maps, multipass	RT	3.3.2
[NHIN86]	/	one	local common illumination	ray casting	NRT	3.3.2
[SSI99]	/	one	global common illumination	ray casting	NRT	3.3.2
[SHC+94]	✓	one	local common illumination	shadow mapping	RT	3.3.3
[HDH03]	✓	one	local common illumination	shadow volumes	RT	3.3.3
[JNP+95]	✓	one	local common illumination	ray tracing	NRT	3.3.3
[GCHH03]	✓	one	global common illumination	shadow mapping	RT	3.3.3
[Deb98]	✓	one	global common illumination	differential rendering + ray tracing	NRT	3.3.3
[GM00]	✓	one	global common illumination	extended shadow mapping	near RT	3.3.3
[FGR93]	✓	one	global common illumination	radiosity + ray casting	NRT	3.3.3
[DRB97]	✓	one	global relighting	hierarchical radiosity	near RT	3.3.3
[LDR00]	✓	one	global relighting	hierarchical radiosity	near RT	3.3.3
[BG01]	✓	one	global inverse illumination	ray tracing	NRT	3.3.3
[LFD+99]	✓	few	global relighting	hierarchical radiosity + ray casting	near RT	3.3.4
[GHH01]	✓	few	global relighting	ray tracing	NRT	3.3.4
[YDMH99]	✓	many	global inverse illumination	ray tracing	NRT	3.3.5
[YM98]	✓	many	global inverse illumination	ray tracing	NRT	3.3.5

Table 3.1: *Overview of illumination methods for mixed reality based on their usage of 3D model, ♯ images, methodology, rendering method and their computation time*

[GM00][GCHH03] and others. The radiance capture calibrates an environment map in the 3D scene, and projects its radiance correctly onto the geometry. Therefore it can be used in environment mapping methods which require well-aligned environment maps to simulate reflections. This would improve the methods described by Sato et al. [SSI99] and Agusanto et al. [ALCS03]. The relighting method in chapter 6 requires the knowledge of the scene geometry and requries two images per scene point for which a reflectance is calculated. These two images are a regular input image and a lightprobe image captured under the same illumination conditions as the input image. The main contribution of the relighting method is that it allows relighting of scenes captured under less restricted conditions, such as under dynamic illumination. This is a major improvement compared to methods presented in this chapter and in particular those presented in sections 3.3.4 and 3.3.5 which require many different input images under controlled illumination.

The inverse illumination method that is required by the relighting method presented in chapter 6 is most closely related to that presented by Boivin et al. [BG01], and provides an improvement to deal with inaccurate geometric reconstructions and more complex scene illumination including dynamic illumination. The BRDF calculation required for the relighting can be time-consuming. Monte Carlo sampling is used to render the relit scene, and therefore the relighting does not operate in real-time. The quality of the BRDF calculation and the relighting are assessed through comparisons with photographic reference images, and through the consistency of the retrieved BRDF for objects made from the same material.

This thesis further focuses on radiance capture and provides an improvement to the state of the art in HDRI generation by allowing the capture of HDRIs in more relaxed circumstances (no use of tripod, object movement allowed). This is interesting for all illumination methods presented in this chapter, but again in particular for those presented in sections 3.3.4 and 3.3.5 which are very sensitive to object movement or deformation, due to their requirement of needing many input images.

3.6 Chapter summary

In this chapter the concept of mixed reality is presented, along with the three types of illumination methods that exist in that context: common illumination, relighting and physically-based illumination. A classification is given of the illumination methods for mixed reality available in the literature. This classification divides the methods into four groups based on the amount of input data required to execute the methods. When comparing the different methods, it is clear that the more input data is available the higher the quality of the resulting MR. The total computation time is proportional to the amount of input data that needs to be processed.

The current state of the art in illumination methods fails to address the issues associated with the capturing of the input data. The problems with an incorrect geometry or radiance capture are either ignored or avoided. Applying the presented methods on uncontrollable scenes that contain object and camera movement and/or are lit by a dynamic light source, will most likely fail to produce a realistic mixed reality. Relying on an accurate geometry and radiance capture reduces the robustness of the illumination methods, and the applicability of these methods to complex-to-model scenes.

Chapter 4

Common illumination with low-detailed geometry

4.1 Introduction

Chapter 3 gives an overview of illumination methods for MR and explains that three different types of illumination simulation algorithms exist: common illumination, relighting and physically-based illumination. To simulate any of the three types of illumination some knowledge about the scene geometry and radiance needs to be at hand. This statement is further supported theoretically in chapter 2.

Common illumination usually operates on a per-pixel basis: the lighting effects are enforced by changing the scene radiance at pixel level. For reasons that become clear later in this chapter, this per-pixel operation is very sensitive to geometric inaccuracies. Inverse illumination usually operates per triangle, thereby effectively averaging errors over groups of pixels making them less obvious to the observer. If the illumination method needs to identify the lighting effects in the real scene texture, the inaccurate geometric reconstruction will fail to deliver the position of the lighting effects accurately. This becomes a problem when new lighting effects of the virtual object need to be simulated over the lighting effects already present in the scene (for common illumination) or, likewise, when they need to be cancelled (for relighting).

This chapter focuses on the shadow simulation in the context of common illumination for scenes with a poorly reconstructed geometric model of the scene, including the scene objects and the positions of the light sources. The contribution of this chapter lies in the method that delivers consistent shadows for MR when only a poor geometric model of the real scene is available. The shadow generation methodology consists of three steps. The first step deals with the correct detection of the lighting effects already present in the scene using image processing techniques. The second and third steps deal with the correct virtual shadow generation through scaling of the scene radiance, regardless of the possibility that there are underlying shadow effects in the scene radiance. The result is a virtual shadow consistent with the real shadows already present in the real scene. The presented methodology is applied to two different types of augmented reality: a pre-computed and a real-time application. Both applications work on real scenes showing *semi-soft* real shadows, which are hard shadows with soft edges.

This chapter is organized as follows. Section 4.2 describes the problems common illumination, shadow generation in particular, faces when the geometric reconstruction is inaccurate. Section 4.3 provides the 3-step methodology that can be applied to generate virtual shadows consistent with the real shadows in the scene. Section 4.4 describes each of these three steps in more detail. This 3-step module is applied to

two different contexts of augmented reality in section 4.5. The limitations of the current implementations of the method are discussed in section 4.6. Finally, section 4.5 gives a summary.

The methodology and generated results have been presented at Graphics Interface 2005 [JAL+05]. The work presented is entirely my own, except where indicated in sections 4.4.3 and 4.5.2. In those sections some implementation issues are discussed which have been implemented with help from Cameron Angus and Jean-Daniel Nahmias.

4.2 Problems with an inaccurate geometric reconstruction

4.2.1 Problem description

For common illumination it is not necessary to know the reflectance of the materials in the real scene to simulate consistent lighting effects. Often the virtual shadows are generated through scaling of the radiance of those scene points that fall inside a virtual shadow. The scaling simulates the blocking of light. A theoretical motivation for this type of shadow simulation has already been given in chapter 2. Often the methods that simulate shadows through scaling take into account whether or not a point is already inside a real shadow. If a scene point lies inside a (hard) real and virtual shadow its radiance should not be scaled, this would simply imply blocking the light twice, resulting in an overly dark virtual shadow.

This is further explained in figure 4.1 (a) through (f), which also defines the terminology used in the remainder of this chapter. Each image shows a graphical interpretation of a real 2D scene in a different context. The real scene is shown in (a) and consists of one object (white rounded box) with a complex shape, a light source (white circle), and a ground plane. The box casts a *real shadow* on the ground plane, which is visualized by the white/grey band. The band is white where light reaches the ground plane and grey where the light is blocked and no direct light reaches the ground plane. Images (b) through (f) show different configurations of an MR set-up. Obtaining the configuration shown in (f) is the objective of this chapter. The top white/grey band (denoted by the letter V, for virtual) shows the shadow configuration brought forward by the particular configuration shown in (b) through (f); the original lighting setting from the configuration in (a) is repeated under the virtual band (denoted by the letter R, for real). Image (b) shows a poor geometric reconstruction of the white box in black which is a simplification of the actual shape of the white object. Image (c) illustrates the reconstruction of the light source position (black circle). For both (b) and (c) it is clear that the position of the *reconstructed real shadow* (or *real shadow estimate*), derived from the reconstructed geometry, is misaligned with the actual real shadow of the white object. Images (d), (e) and (f) show the inclusion of a virtual object (dark shade of grey). In (d) the shadow of this virtual object is simulated through scaling the pixels that fall inside the *virtual shadow*, derived from the reconstructed light source position and the geometry of the virtual object. The pixels that lie inside both real and virtual shadows are too dark; this is due to the double scaling. Image (e) shows how the virtual shadow is improved by not scaling the pixels that also lie inside the reconstructed real shadow. However, for those pixels where the real shadow does not overlap with the reconstructed real shadow, the double scaling is still enforced, resulting in an incorrect shadow pattern. Image (f) illustrates the virtual shadow when all pixels that lie inside a real and virtual shadow are withheld from the scaling. The orientation of the virtual shadow and its appearance are consistent with the shadows already present in the real scene.

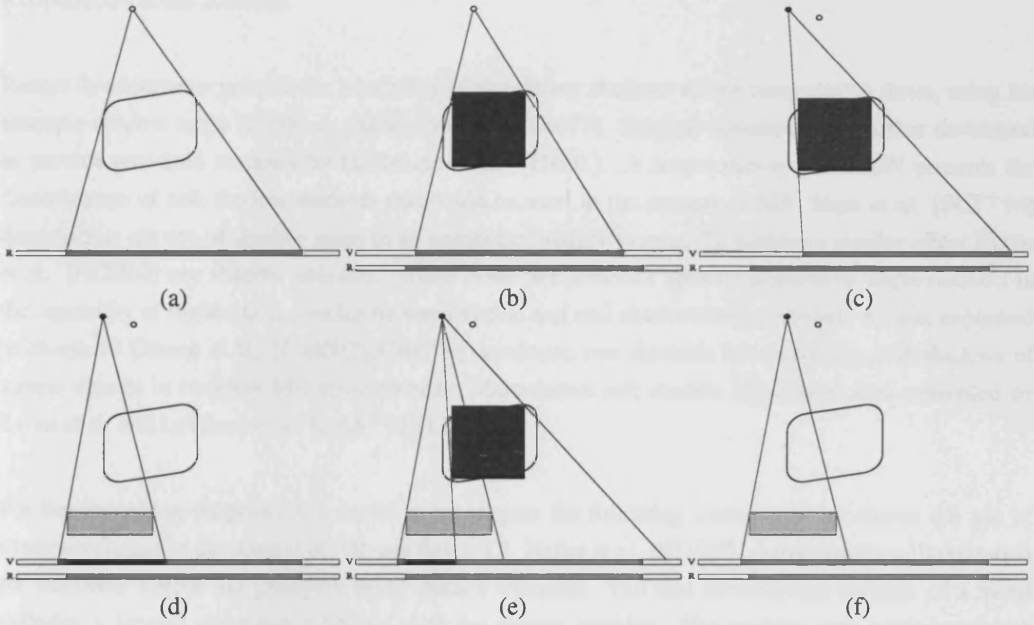


Figure 4.1: *Common illumination simulates the virtual shadows of the virtual objects inserted into the real scene. This is often achieved through scaling the pixels that fall inside a virtual shadow to simulate the blocking of light. Images (a) through (f) show a 2D impression of a real scene in different MR contexts. The white object, the light source indicated by a white circle, and the ground plane are part of the real scene. The black box is the inaccurate reconstruction or geometric model of the white object; the black circle illustrates the inaccurate virtual light source positioning. The white/grey band at the bottom shows the shadow pattern on the ground plane for the real scene, denoted by letter R in (a) and repeated in images (b) through (f), and for the different virtual configurations in images (b) through (f) denoted by letter V. The grey object is a virtual object included into the real scene. The objective is to generate consistent shadows for this grey object in order to simulate common illumination. (b) shows the reconstructed real shadow in the virtual band when an incorrect geometric model is used to represent the white object. (c) shows the reconstructed real shadow (or real shadow estimate) when both geometric reconstruction and light source position are inaccurate. (d) illustrates the inclusion of the virtual object; the virtual shadow is simulated through scaling of the pixel values that lie inside the virtual shadow. Pixels that lie inside real and virtual shadows are double scaled and as a result appear too dark. (e) illustrates that when using the reconstructed real shadow position from (c) to identify which pixels fall inside both real and virtual shadows, the mismatch between reconstructed and real shadows still result in an incorrect shadow pattern. Finally (f) shows how the actual virtual shadow should be simulated.*

4.2.2 Shadows for common illumination

Several methods providing common illumination were already mentioned in chapter 3. Common illumination basically simulates the lighting effects induced by the virtual object onto its real environment through the gathering of light and scaling it by the appropriate material properties. Several methods exist that provide real-time rendering [BGWK03][ALCS03][HDH03][GCHH03], but to our knowledge no real-time illumination method exists that guarantees a consistent shadow generation, regardless of the reconstructed scene accuracy.

Recent developments provide the possibility of simulating shadows at low computation times, using for example shadow maps [Bli88] or shadow volumes [Cro77]. Shadow volumes were further developed to provide real-time shadows by Heidmann et al. [Hei91]. A recent survey [HLHS03] presents the classification of soft shadow methods that could be used in the context of MR. State et al. [SCT+94] demonstrate the use of shadow maps in an augmented reality system. To achieve a similar effect Haller et al. [HDH03] use shadow volumes. While these two methods showed impressive improvements in the capability of registration, overlap between virtual and real shadows were avoided. As was explained in chapter 3 Gibson et al. [GM00][GCHH03] developed two methods for simulating soft shadows of virtual objects in complex MR environments. More recent soft shadow algorithms were presented by Laine et al. and Lehtinen et al. [LAA+05][LLA06].

For benchmarking purposes it is useful to investigate the following example which shows the use of shadow volumes in the context of AR, see figure 4.2. Haller et al. [HDH03] create common illumination for relatively simple 3D geometry using shadow volumes. The real environment consists of a metal cylinder, a ground plane and a fiducial used for camera tracking. The tracking was carried out with ARToolkit [KBBM99]. The shadow cast by the real object is a hard shadow, indicating that a point light source was inserted into the real environment. The virtual objects are the pink/blue torus and sphere. Shadow volume methods proceed by identifying the position of the virtual shadow from the current viewpoint, using the stencil buffer, and scaling the radiance inside this area to simulate the blocking of light. The results for Haller et al. are shown in figure 4.2. Images (a) and (b) show how the virtual objects can cast shadows onto the metal cylinder, the ground plane and onto each other. The shadows look consistent, as the choice of scaling factors used to scale the radiance within the virtual shadows is appropriate. When the virtual shadow overlaps with a real shadow, as is shown in (c) and (d), the scaling of the pixels inside the virtual shadows seems inappropriate. The radiance inside both the real shadow and the virtual shadow is scaled too much. In fact it is scaled twice: once due to the real shadow, and once due to the virtual shadow, which makes the shadow look too dark. This is because Haller et al. made no effort to prevent the pixels inside the real shadow from being scaled.

Figure 4.3 illustrates how the overlap of two real shadows usually remains unnoticeable in a real environment. In this example a strong light source casts a strong (semi-soft) shadow on the bookshelf. Images (a) and (b) illustrate where the shadow, highlighted in grey, is cast due to the support plane of the shelf (a) and the CD box (b) which are highlighted in orange. The two objects cast overlapping shadows. Image (c) illustrates that from the intensities of the shadow it is impossible to say which part of the shadow belongs to the shadow cast by the support plane or the CD box. In other words, the overlapping of the shadows is not noticeable from the shadows alone, unlike the incorrect virtual shadow simulation shown in figure 4.2 (c) and (d).

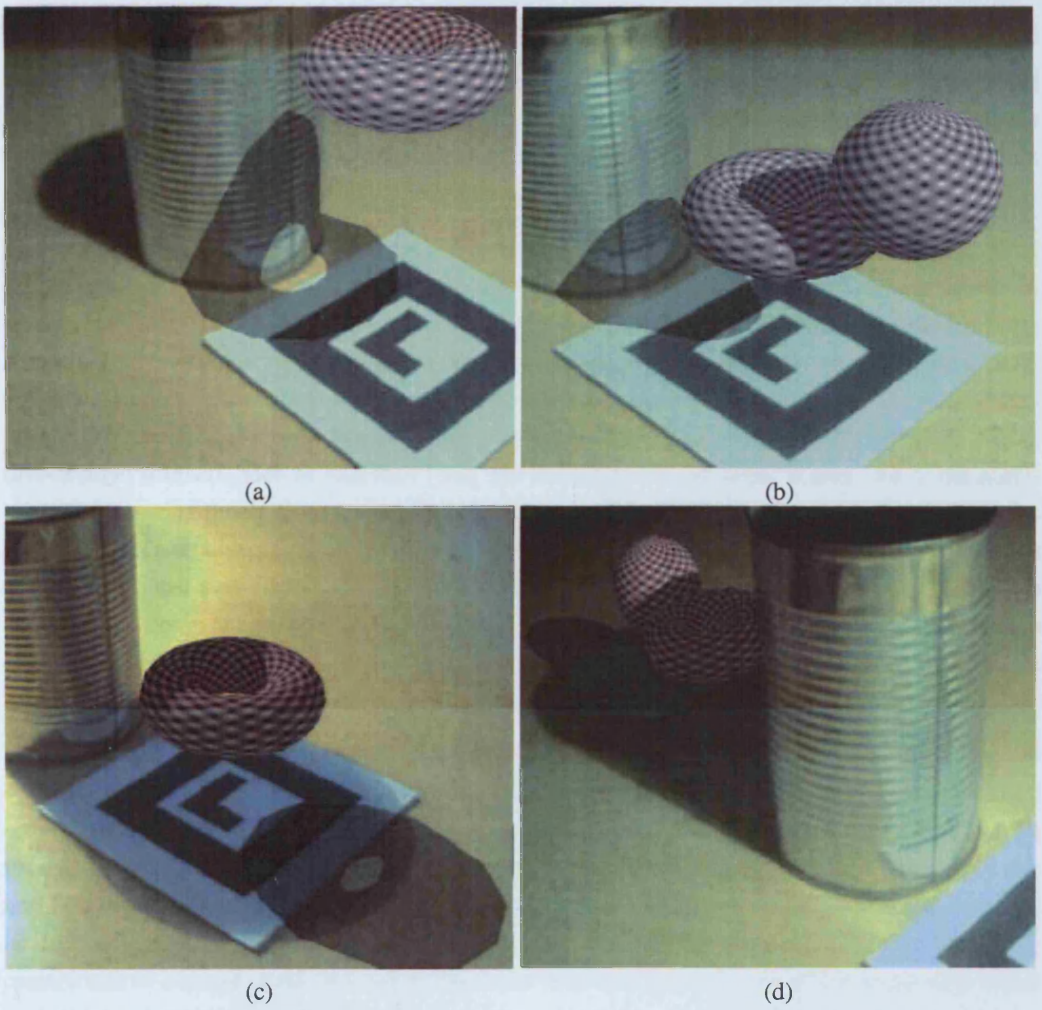


Figure 4.2: *Haller et al. [HDH03] provide common illumination for augmented reality using shadow volumes. The real environment consists of a ground plane, a metal cylinder and a fiducial to aid the camera tracking. The pink/blue torus and the sphere are virtual objects for which the shadows are simulated. (a) The virtual object casts a shadow on the cylinder and the ground plane. (b) The shadow from the sphere is cast onto the torus. (c) The virtual shadow overlaps with the cylinder's real shadow and the image shows how this results in inconsistencies. (d) Again, the virtual shadows overlap with the real shadow. Due to the double scaling, the shadow is too dark and inconsistent with the other shadows in the real scene. Courtesy of Haller et al.*

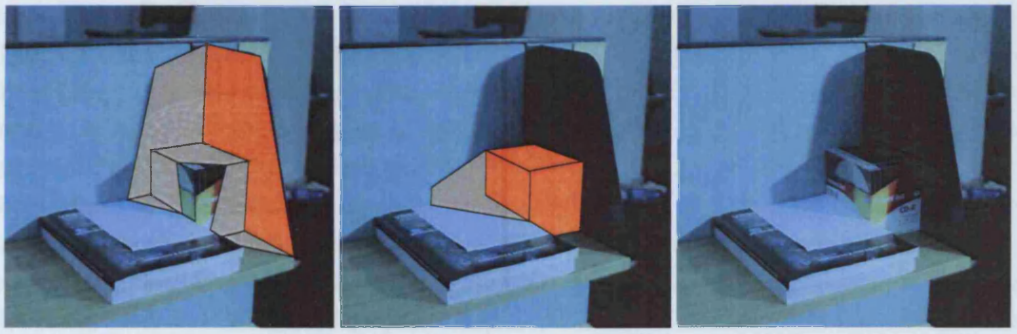


Figure 4.3: *When shadows overlap, border effects are usually not visible. (a) The right (support) plane of the shelf (shown in orange) casts a shadow (shown in grey) on the bottom plane of the shelf and the blue divider. (b) The CD box (in orange) also casts a shadow (shown in grey) on the bottom plane of the shelf and the blue divider. (c) The shadow does not show any boundary effects due to the overlapping of the two shadows as the strong light source casts strong (semi-soft) shadows.*

4.2.3 Reconstruction inaccuracies

When considering the shadow generation as outlined in figure 4.1, the ideal reconstruction of a 3D scene is one in which the retrieval of the lighting effects does not result in visible artefacts during the shadow simulation. Before finding solutions for the problems that arise with an inaccurate geometric reconstruction, we need to investigate how inaccurate a geometric reconstruction actually can be.

The accuracy obtained with 3D laser scanners is high as long as the scanned object/environment obeys certain rules. Scanners are particularly sensitive to dynamic deformations of the object and vibrations of the scanner or object. The accuracy can also be compromised by the material of the objects being scanned. This problem is frequently met in cultural heritage, when for instance a marble statue is being scanned. The sub-scattering properties of marble create inaccuracies [GBR+01]. All this makes scanning larger scenes error-prone. Boehler et al. presented a survey on laser accuracy [BM03]. Active *time-of-flight* scanners, such as 3RDTech [3rd], are unsuitable to scan larger scenes, due to the uncertainty in measuring time[1]. Other types of scanners like structured light scanners[2], such as Camtronics [Cam], can scan an object quickly compared to other types, but are less suitable for larger scenes due to ambiguities that may occur[3]. 3D scanners provide a set of scene points, usually triangulated into a mesh which forms the scene surface. The problem with such a scene description is that there is no object information. A segmentation of the scene into objects might be required in inverse illumination calculations.

The cost of using a 3D scanner is much larger than that of user-aided reconstruction software such as ImageModeler [Rea] and PhotoModeler [Eos]. The time it takes to capture a 3D scene with reconstruction software is usually shorter than the time it takes to capture a scene with a scanner. For reconstruction systems, it suffices to capture several images from different viewpoints. It is the post-processing (the actual modelling) time that makes reconstruction software less attractive. Some research work focuses on decreasing the modelling time, for instance, Freeman et al. [RFZ05] allow rapidly rudimentary reconstruction of a simple scene. It is fair to argue that reconstruction software is less sensitive to object movement and deformation than reconstruction using scanners. After all, the user can choose which points are used to reconstruct an object from. From a practical point of view it is reasonable to state that carrying around a 3D scanner is by far more difficult than capturing a scene with a camera.

We should also not ignore the fact that, when not used in the context of illumination methods, the accuracy of a reconstruction is visually less important when appropriate textures are mapped onto the 3D model. The sense of shape is often created, and enhanced, by appropriate texturing. The textures contain cues of depth through occlusions, highlights and shadows that are visible. This is illustrated in figure 4.4 where the left column shows images of a textured model of a laptop and the right column shows images of the un-textured geometric model. The fiducial visible in texture of the laptop is used to track the camera in one of the applications presented in section 4.5.2 of this chapter. The texture of the keyboard contains many highlights and shadows, the occlusions of the different keys provide significant depth information. The back of the laptop contains textured sockets, which again due to occlusion effects create some depth perception. The un-textured model is rendered from the same viewpoints, under the same illumination. Due to the resulting shadow effects, there is still a sense of depth, but it is clear that the *keyboard* and back of the laptop consist of simple triangles. The statement that textures create

[1] A Time of flight scanner measures the return time for a laser dot. As an example, one millimetre difference gives a difference of 3.3. *pico*seconds of return time.

[2] A grid is projected onto the object, from the deformation of the grid, depth is calculated. Using a grid improves the scan time, as larger portion of the scene can be scanned at the same time.

[3] The lines in the grid, are not necessarily perceived in the same order as it is projected, there is no way to know which perceived line belongs to what line projected.

a sense of depth is further supported in the literature, psychological research on human perception in a natural environment indicates that humans derive depth from several cues [HH73][MR87]: occlusions, binocular disparity, perspective effects, movement, shadows and textures.

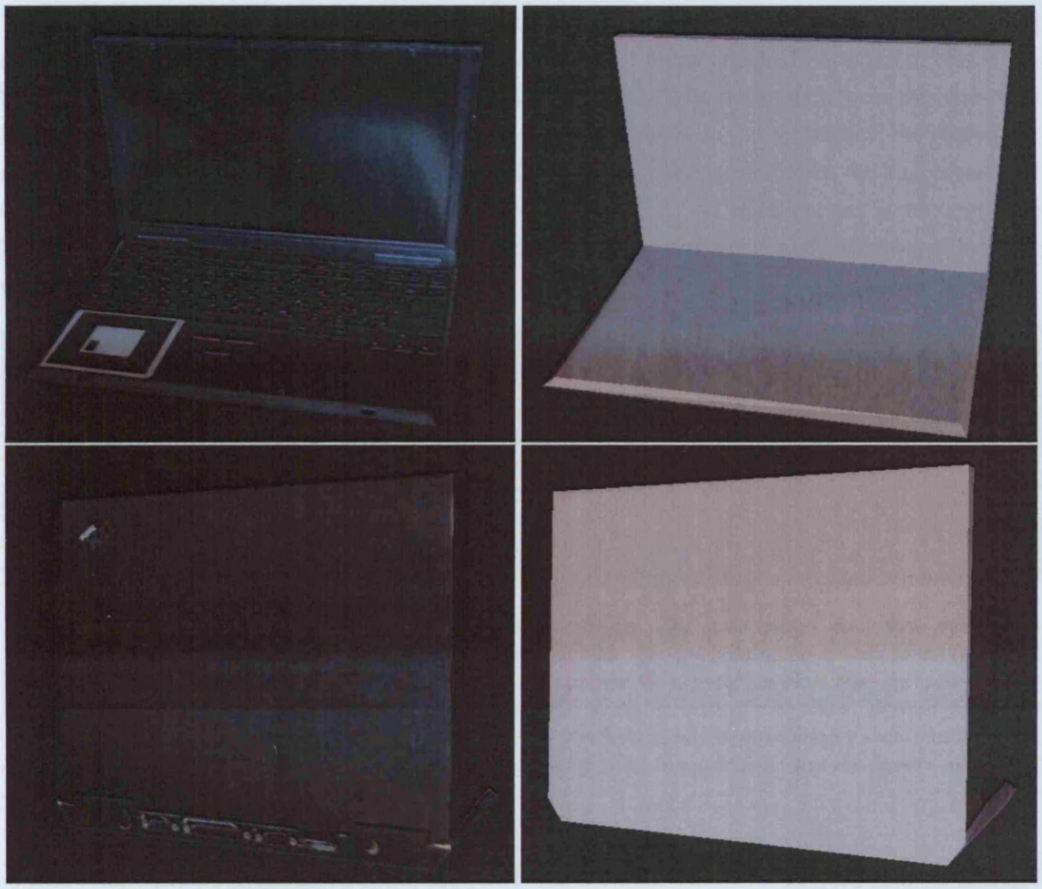


Figure 4.4: *Textures projected onto the geometry produce phantom shapes. The images in the left column show a textured model of a laptop from two different viewpoints. The images in the right column show the geometry of the laptop model from exactly the same viewpoint. While the model clearly consists of triangles (see right column), the textures create a notion of shape, e.g., the keyboard and sockets at the back of the laptop create depth (see left column). This hint of depth information is created through the shadows and highlights visible in the textures.*

It should be noted however, that if the shadows visible in the texture need to be removed (e.g., in relighting applications), it will be important to have an accurate geometric model. The model shown in figure 4.4 would in general be insufficient to remove the shadow effects on the keyboard using the methods described in chapter 3. Also, when a virtual shadow is projected onto the keyboard, its shape will be incorrect as the local detail of the objects (keyboard) is not known. In other words, the requirement for an accurate reconstruction might not be necessary for aesthetical reasons, as long as an appropriate texture is mapped onto the geometry, but the current state of the art in illumination generation will most likely fail to generate consistent lighting effects. However, efforts to produce accurate geometric models using expensive equipment can actually be seriously reduced, as long as solutions can be found to compensate for the lack of detailed geometry.

When simulating common illumination for augmented reality the shadow detection is not only influenced by the accuracy of the geometric model, but also by the accuracy of the tracking algorithm used. The

performance of registration tools like ARToolkit, depends on how well the camera sees the fiducial used to register the camera [Mel95][NC96]. Highlights and shadows visible on the fiducial limit its usability in a real environment.

This inaccuracy due to the camera tracking has been experienced during several experiments. Figure 4.5 shows three frames of video footage, showing a laptop with a fiducial attached. Overlaid in blue is the 3D model of the laptop, which consists of two planes, one aligned with the laptop screen, the other with its base. The registration of the blue model with the laptop depends on the accuracy of the model, but also on the accuracy of the camera tracking using the fiducial. In the three frames, the blue model and the actual laptop are clearly misaligned. When the camera moves, the misalignment is not consistent between different frames; this further indicates that the misalignment is due to the tracking rather than the inaccuracies in the model.

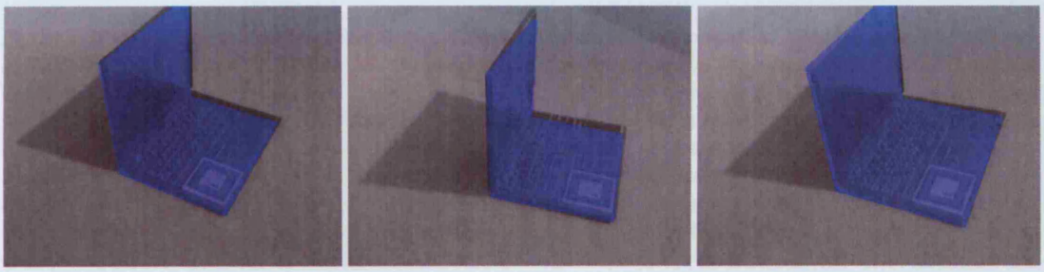
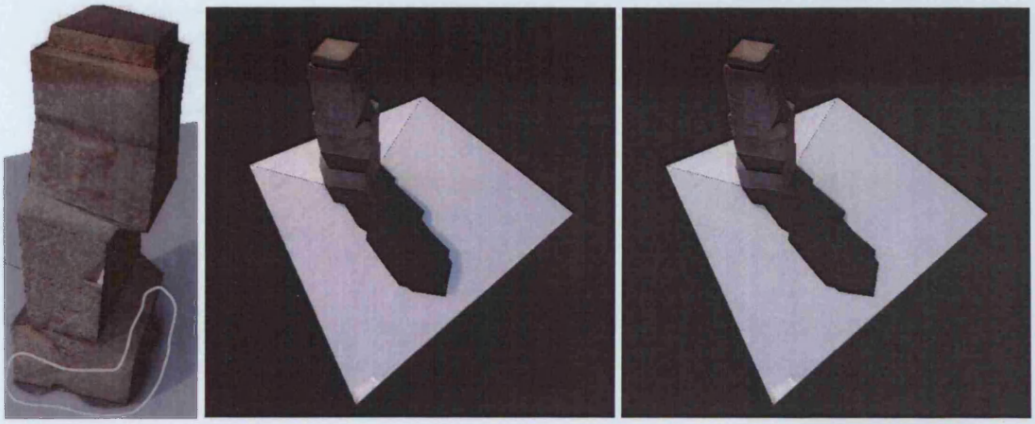


Figure 4.5: *Augmented reality: inaccuracies in registration software. The three images show three frames of a video captured with a camera. The camera tracking uses ARToolkit. The position of the camera to the fiducial is calculation. When this position is known, a model of the laptop can be overlaid (in blue) over the video footage using the (known) position of the model from the fiducial. The model consists of two planes, derived from a 3D model of the laptop. These planes should align perfectly with the laptop, but the three images show that there is a clear mismatch. The misalignment is not static throughout the frames, in particular when the camera moves and causes registration jitter.*

Texture extraction and mapping onto the reconstructed geometry is another source of errors. When the camera calibration is inaccurate, texture extraction and mapping is error-prone. This is illustrated in figure 4.6 (a), where part of the statue's texture is mapped onto the ground plane. The mismatch between geometry and texture might be small and negligible, but even these small misalignments can result in significant artefacts when simulating virtual shadows over real shadows.

Finally we also need to mention the influence of the accuracy of the light source position on the estimation of the real shadow estimates. Estimating the position of the light source position in an indoor, controllable environment is fairly easy as the light source can be measured and manipulated. Finding the position of the light sources in an outdoor, uncontrollable environment is less straightforward. While the time of the day, the weather conditions and the global position of the objects on earth can be used to calculate the light distribution [DK02][PSS99], it is cumbersome to actually look up the coordinates and orientation of the scene objects. It would be nice if the requirements for the light source position were less stringent. The influence of the position on the real shadow position is nevertheless significant. Figure 4.6 (b) and (c) are rendered from the same viewpoint. Each image shows the same 3D textured model, consisting of a statue placed on a table cloth. A semi-soft shadow (lighter grey) due to the sun, is visible on the cloth. Based on an estimate of the light source position and the 3D geometry, the real shadow is reconstructed. The reconstructed real shadow (dark grey for illustrative purposes) is projected over the textured 3D model. In (b) the reconstructed shadow and the real shadow do not overlap very well. After manually re-positioning the light source to maximize the overlap, we can still see in (c) that not all parts of the real shadow are covered by the reconstructed shadow. The difference between the



(a) (b) (c)

Figure 4.6: *(a) Inaccurate camera calibration results in misalignment between real and virtual model. The influence of the light source position on the detection of the real shadow positions cannot be underestimated. The 3D textured model shown in (b) and (c) consists of a statue positioned on table cloth, casting a semi-soft shadow (lighter grey) on the cloth due to the sun, see also figure 1.6. Overlaid in dark grey is the reconstructed real shadow, based on the position of the light source and the 3D geometry. There is a clear mismatch between the dark and light grey shadows in (b), this is due to a slightly wrong light source position. After manually aligning the dark and light grey shadows, the mismatch is seriously reduced (c), nevertheless the reconstructed shadow still does not overlap the real shadow due to the incorrect geometry. The angle between the light direction in (b) and (c) (relative to the centre of the modelled scene) is only 2.6 degrees.*

directions of the infinite light source in (b) and (c) is only 2.6 degrees. This actually highlights another problem: when a scene is captured over a long time interval and the light position changes (e.g., the sun) the shadows visible in the texture for different objects, may have a different orientation. The pre-defined light source position may correctly retrieve the position of the shadow for one object, but may fail to find correctly the position of another object.

4.3 Methodology

The previous sections outlined that the geometry capture can be insufficient for common illumination methods that apply lighting effects over the captured radiance distribution. It was illustrated that common illumination, generated by applying a scaling on the radiance texture, returns inconsistencies when virtual shadows overlap with real shadows, even when the geometric model is used to detect the real shadow prior to applying the scaling to prevent the scaling of the pixels inside a real shadow. In this section a novel shadow generation algorithm is presented that generates consistent shadows for poorly reconstructed scenes, in an AR context. Our implementation is limited to scenes containing *semi-soft shadows*. Semi-soft shadows are sharp shadows with a certain (limited) degree of softness at their borders. A semi-soft shadow has mainly umbra[4] and only partially penumbra characteristics.

The methodology consists of three steps, illustrated in figure 4.7, and can be easily incorporated into any type of common illumination algorithm that applies a scaling to simulate shadows in a real environment:

1. **Shadow detection:** this step deals with the detection of the position and shape of the real shadows visible in the radiance texture. This step takes as input the geometric reconstruction of the real scene, the scene radiance, and the position of the light sources. Once the true shadow positions are known, it is possible to calculate a scaling factor, called the *shadow factor*, for each material

[4] A shadow usually consists of umbra and penumbra areas. Points that are completely blocked from the (direct) light sources lie inside umbra; points that are partially blocked from (direct) light sources lie inside penumbra.

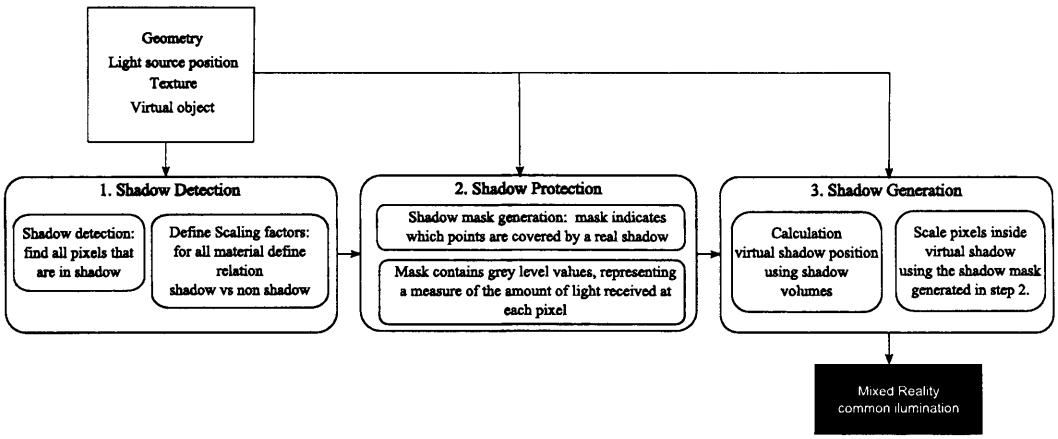


Figure 4.7: *Consistent shadow generation using a 3-step methodology. The three steps are: shadow detection, shadow protection, and shadow generation, and are indicated by rounded grey boxes. These steps rely on the following input: a geometric model of the scene, an estimation of the light source position, the scene radiance in the form of a texture, and the virtual object that is included into the real scene. The output of the total process is a mixed reality with local common illumination, indicated by the black square box.*

in shadow. This factor relates the colour for shaded and non shaded regions for a specific material. A more detailed explanation of the shadow detection step is given in section 4.4.1.

2. **Shadow protection**: a *shadow mask* is created and used to protect the points inside a real shadow from scaling during the virtual shadow simulation. The shadow mask can be mapped onto the scene geometry, and as such, defines the scaling for all scene points. The shadow mask is binary when the scene consists of hard shadows where a "1" allows scaling and a "0" prevents scaling. The shadow mask contains grey-level values when semi-soft shadows are considered. In that case, the grey value is a measure for the amount of scaling that is allowed for a certain scene point. More details about the shadow protection step are given in section 4.4.2.

3. **Shadow generation:** a real-time shadow method such as shadow maps or shadow volumes is used to generate the virtual shadows. The colour of a virtual shadow is defined by the appropriate shadow factor computed in step 1. Overlap between real and virtual shadows is prevented by using the shadow mask generated in step 2. Section 4.4.3 provides more details about this third step.

The methodology as outlined above focuses on the shadow generation for common illumination, but it is important to note that the framework can also be used to simulate other lighting effects, such as highlights, as long as there is a suitable method at hand that can detect such lighting effects. This framework can even be used in relighting applications which usually removes the current lighting effects in the scene before applying the new lighting effects due to a different illumination setting. The lighting removal will fail if the geometry is not accurately aligned with the scene radiance due to for instance an incorrect 3D model. To compensate for the resulting inconsistencies a slightly modified 3-step methodology can be developed. The shadow detection step can be used to detect the position of the shadow in the real scene texture. The shadow protection step can then be substituted by a shadow removal step, to ensure all shadow pixels are effectively *de-shadowed*. The final step can then be used to apply the new shadows in the real scene, based on the novel illumination settings.

4.4 Generating consistent shadows

The following sections give a more detailed explanation of the different steps presented in the previous section and illustrated in figure 4.7. In chronological order these steps are: shadow detection, shadow protection and shadow generation.

4.4.1 Shadow detection

A Shadow contour detection

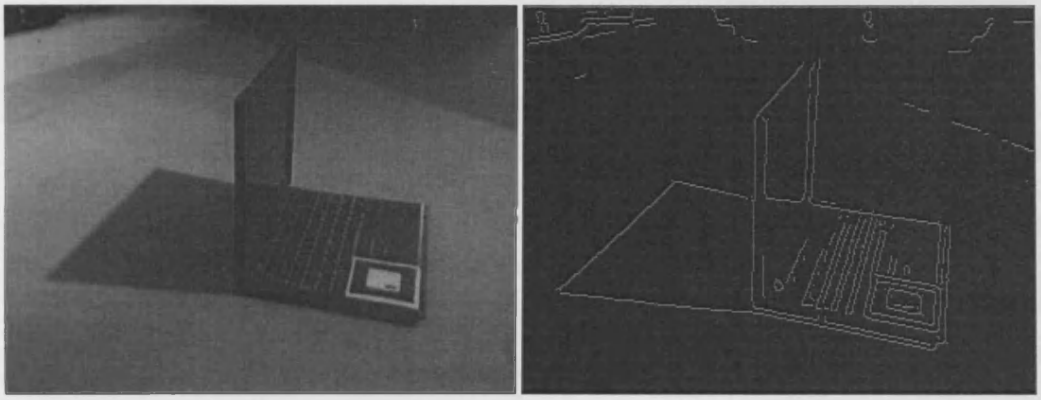
In order to protect the existing shadows in the scene from any post-processing, the shadow pixels in the texture need to be identified. Different types of shadows exist and not all shadow detectors are suitable for each type of shadow [PMGT01]. Detecting semi-soft shadows is usually easier than detecting soft shadows, as semi-soft shadows contain more feature (edges, contrast) compared to soft shadows, which makes the latter more difficult to detect. As it is beyond the scope of this thesis to build a new shadow detector, we have chosen to implement a semi-soft shadow detector, which succeeds in detecting a range of shadows from purely hard shadows to semi-soft shadows. This is further motivated by the fact that the 3-step methodology is actually irrelevant of the choice of shadow detector. Hard shadows are created when the only light source in the scene is a point light source, and therefore cannot exist in a real-world environment. However, semi-soft or well-defined shadows appear in scenes with bright light sources, e.g. sunny outdoor scenes, and indoor scenes with one main light source. The presented shadow detector has been tested in both types of scenes with good results. It should be stressed that if a soft shadow detector is used the method can be applied on less restricted scenes.

A Canny edge detector [Can86] is implemented to detect shadow contours in a similar approach as in [Bev03][SGE01][SCE04]. The main advantage of the Canny edge detector is that it returns one-pixel thick edges and that it effectively neglects edges due to noise because of the non-maximal suppression and hysteresis. Our shadow detector uses a geometric estimate of the shadow contour to generate the contour of the real shadow. The geometric estimate of the shadow contour is derived from the real shadow estimate, which is calculated from the approximated geometric reconstruction and the virtual light source position. In general the shadow estimate gives a good indication of the position of the real shadow, but it can be inaccurate due to the error of the estimated light source position within the scene and the geometric approximation of the real scene geometry, as was discussed previously. In our implementation, a bounding box is constructed around the real shadow estimate. The Canny edge detector then operates inside this bounding box. An example of input and output of the Canny edge detector, applied to a scene showing semi-soft shadows, is given in figure 4.8.

The edge detector operates on 2D images or image frames. This 2D image can be derived from one of the three colour channels or from the intensity of the pixels, where the intensity of a pixel can be calculated as the average of the three colour channels. If only operating on one colour channel, the user needs to be careful that this colour channel shows all scene features clearly.

Usually an edge detector can only detect hard shadows and gives a rough estimate of a soft shadow contour as long as the gradient of the soft edge is larger than the gradient of any other underlying structures in the texture. When the parameters of the edge detector are tuned to detect the soft edges, the amount of false edges due to noise for instance, increases as well. Our shadow edge detector guarantees a correct shadow contour detection when:

1. The geometric centre of the real shadow estimate is near to the geometric shadow of the real shadow, in other words, the real shadow must lie inside the bounding box. This assumption makes



(a) (b)

Figure 4.8: *(a) An input image, showing a laptop casting a clear semi-soft shadow on the ground floor, is fed into the shadow detection system. (b) The Canny Edge detection finds the shadow contour (white) among other edges clearly visible in the input image.*

abstraction of the shape or orientation of the real shadow estimate, it suffices to have an estimate that is *similar* in position, size, and orientation.

2. The shadow is a semi-soft shadow: which is a soft shadow but with a small penumbra area. This ensures that the detected shadow contour is in fact a shadow border. The Canny edge detector would fail to detect truly soft shadow borders.

3. When the texture contains a pattern (e.g., stripes, dots, etc.), the contrast at the shadow edge needs to be larger than the contrast of the pattern. When this is the case, an appropriate choice of the smoothing filter's size and the upper and lower thresholds of the Canny edge detector, will suppress the edges of the background texture against the edges of the shadow.

The computation speed of the shadow edge detector depends on the resolution of the input image. The most time-consuming operations are the smoothing and gradient convolutions, which can be sped-up when implemented as multiplications in the Fourier domain. However, a transformation to the Fourier domain was not implemented since other, more intuitive, time reducing steps can be taken. Firstly, the edge detection only needs to operate in the bounding box around the real shadow estimate. Secondly, the search region can be further reduced to the areas around the virtual shadow, as it is only necessary to know the real shadows in those regions. In other words, with our implementation, the computation speed rather depends on the size of the real and virtual shadows, than on the size of the textures. In all experiments carried out, the shadow detection operated in real time (around 30 frames per second).

B Defining the relationship between pixels inside and outside a shadow region

The pixel values inside a shadow are always darker than the pixel values outside a shadow, as long as these pixels cover the same material. Research carried out by Cavanagh et al. [CL89] showed that to simulate the impression of a shadow, the exact scaling factor is not as important to identify a shadow region from a non-shadow region, more important is that the shadow region is darker than the non-shadow region. When two shadows are cast on the same material, however, it is important that the colour of the two shadows are similar (variation in colour might exist due to the orientation of the material to the light source and to the observer or due to local differences in BRDF). When simulating virtual shadows, it is therefore necessary to create shadows consistent with the real shadows of the scene. This

is approximated in our implementation by calculating a shadow factor per material in the scene, which relates shadow regions with non-shadow regions for a certain material.

It is possible to calculate the shadow factor once the shadow contour/region is known. The shadow factor is in fact a triplet $[\rho_R, \rho_G, \rho_B]$, defined by dividing the average of the three colour channels over a set of pixels inside the shadow by the average of the colour channels over a set of pixels outside the shadow. In our implementation, a small region of pixels inside and a small region of pixels outside (but near to) the shadow are selected and used to derive an average shadow factor per colour channel. The following equation formulates mathematically how the triplet is calculated. $C \in \{R, G, B\}$ represents the colour channel: red, green or blue; SR stands for shadow region, while NSR stands for non-shadow region; P_{SR} and P_{NSR} are the number of pixels in respectively the shadow and non-shadow region:

$$\rho_C = \frac{\frac{\sum_{\forall p \in SR} C_p}{P_{SR}}}{\frac{\sum_{\forall p' \in NSR} C_{p'}}{P_{NSR}}}$$

In general the shadow factor should vary across the points on a surface, as it depends on the orientation of the surface to the light source, the area of the light source, and some local material differences. It would be better to assign a scaling factor per pixel, however only one scaling factor is used per material for the applications developed in this chapter as it proved to return acceptable results.

Figure 4.9 gives an overview of how the shadow factors are defined in the current implementation. In this figure a white object casts a shadow onto two different materials (the ground plane and the side wall). A, B, C and D are the average colour of the material in small regions indicated in the figure. For each material there is a different scaling factor, for the ground plane it is given by the ratio $\frac{B}{A}$ and for the side wall it is given by the ratio $\frac{C}{D}$. In the current implementation, the scaling factors were defined manually, but in principle, once the shadow region is known, it is possible to select a local area outside the shadow and a local area inside the shadow automatically. We usually selected small areas with an approximated size of 20×20 pixels on images with dimensions 1536×1024. If this did not resulted in a satisfying virtual shadow colour, a slider allowed user intervention to select a more appropriate scaling factor. Our choice of scaling factor rarely failed, but it did produce noticeable incorrect shadow factors when the texture contained specular highlights or when the selection window used was not representative for the remainder of the texture in general. The calculation of the shadow factor has also been calculated in the HSI colour space, but this implementation did not resulted in a higher performance of the shadow factor calculation and failed in similar circumstances as the RGB-implementation, due to the neglected influence of the viewing angle and small local material difference that might be present.

4.4.2 Shadow protection

A Shadow mask generation

Based on the shadow contour, a shadow mask is constructed. This mask is used to indicate which points in the real model are in shadow or not. In a way, it can be considered as a texture map that overlays the original radiance textures of the scene. The shadow mask is defined as follows. The output of the shadow detector is an edge image, containing information about the shadow contour at pixel level (see figure 4.8 (b)). Since the Canny edge detector detects edges of one-pixel thickness and removes most noise inside the image, the shadow region can be derived relatively easy: the shadow mask construction takes the edge map as input and uses a region growing algorithm to fill the shadow region inside the shadow contour. The starting point of the region growing can be any point inside the shadow region, which can be derived from the real shadow estimate. For practical reasons, in the current implementation of the

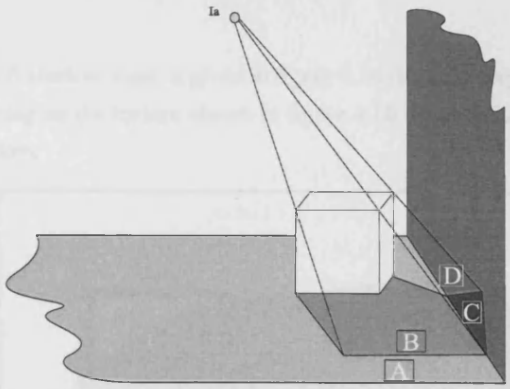


Figure 4.9: *Shadow volumes: finding the correct scaling factor. The white box casts a shadow onto the ground plane and side wall. The ground plane and side wall are made from a different material. Due to the direction of the light relative to the plane on which the shadow is cast, and the type of underlying material, the scaling factor between shadow and non-shadow area is different for the ground plane and side wall. For the ground plane it is given by the fraction* $\frac{B}{A}$*; for the side wall it is given by the fraction* $\frac{C}{D}$*.*

algorithm, the centre of the bounding box is chosen as seed pixel for the region growing. The limitations of this choice and some improvements are discussed later. The binary shadow mask obtained as such, is "0" in shadow areas (no scaling is allowed), and is "1" in non-shadowed areas (scaling is allowed).

The region growing algorithm grows from the seed pixel until it meets an edge pixel. Because it is possible that the shadow contour is not a closed curve, the growing process also stops if it meets two open edge pixels *near* to each other. With near meaning that the distance D between the two open edge pixels is small. The choice of the pixel distance D depends on how well-defined the shadow contour is, a value of 3-5 pixels for D was suitable in our implementation.

Using region growing as defined above to build the shadow mask is not flawless, some false positives (pixels not inside a shadow detected as being in shadow) and false negatives (pixels inside a shadow detected as not being in shadow) can be the result. But our experiments showed that this is a fairly robust implementation.

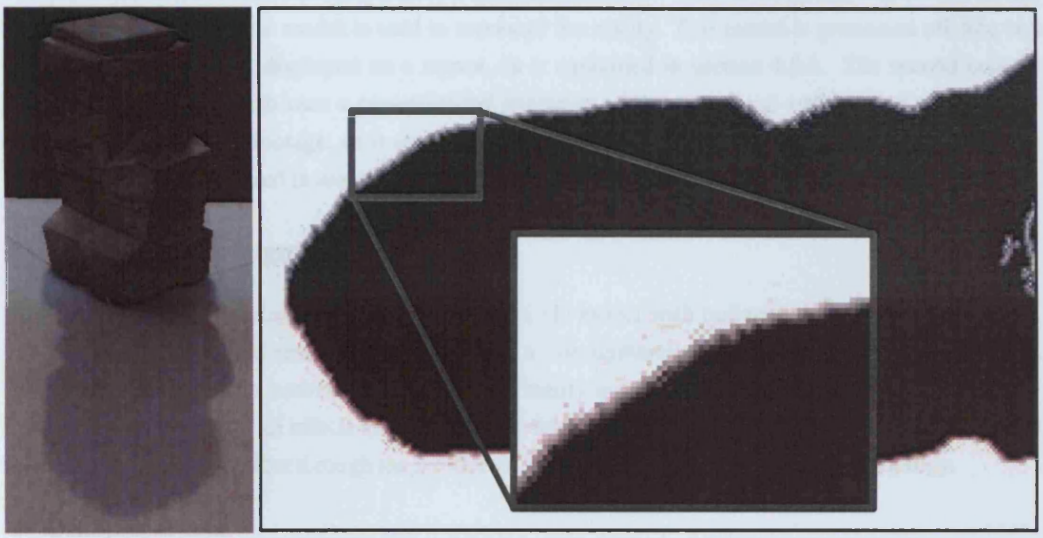
B Semi-soft shadow mask generation

The eventual colour of the virtual shadow is now defined by the multiplication of the underlying radiance texture value times the entry in the shadow mask and the shadow factor. The shadow mask contains discontinuities at the shadow borders, which results in inconsistencies when a virtual shadow is simulated over the shadow borders because the real shadows will in general not show such discontinuities. A better implementation of the shadow mask will allow the scaling factor to decrease gradually near the borders of a shadow.

In this chapter this is obtained by applying a smooth gradient on the values in the shadow mask at the shadow borders, resulting in a semi-soft shadow mask. In this thesis the gradient is calculated using a gaussian smoothing filter. The support of the gradient increases with the *degree of softness*. This can be estimated if the distance of the light source and its area are approximately known. For instance, for an outdoor scene, on a sunny day it is known that the sun has a solid angle of 0.5 degrees. Together with the geometry estimate and the direction of the sunlight an approximation of the degree of softness can be calculated. In the other cases, one can use the behaviour of the pixel intensities around the detected shadow edge to calculate the degree of softness. In this thesis however, the degree of softness has been fine-tuned manually. The improvement to find the degree of softness automatically has been left as future

work.

An example of the semi-soft shadow mask is given in figure 4.10 (b), resulting from applying the shadow detection and region growing on the texture shown in figure 4.10 (a) and after applying the gradient on the boundaries of the shadow.



(a) (b)

Figure 4.10: *(a) Original real scene shows a statue casting a semi-soft shadow on a white cloth. (b) A semi-soft shadow map is generated which covers all shadow pixels shown in (a).*

4.4.3 Shadow generation

Once the shadow mask and shadow factor are generated, it is possible to simulate the virtual shadows. These shadows can be generated with different algorithms. In our implementation shadow volumes are used, but any other real-time shadow generation procedure, including soft shadow generation, is suitable. The shadows are computed using the approximate position of the real light source and considers the interactions between virtual and real objects. Shadows from virtual objects are therefore cast on real and virtual objects, and real objects can cast shadows on virtual objects. Shadows due to real objects onto other real objects are already included in the texture.

In order to prevent any overlap between the generated shadows and the real shadows, the semi-soft shadow mask computed in section 4.4.2 is used to indicate which pixels can be scaled, and how much scaling is allowed. We use the shadow mask in two different ways. When a textured model of the real scene is available the shadow factor is incorporated into the semi-soft shadow mask. Then the multi-texturing capability of OpenGL is used to multiply the original texture and the semi-soft shadow mask texture. When ARToolKit is used, we can draw the shadow mask directly in the stencil buffer. In this implementation each (visible) scene point needs to have one entry in the shadow mask.

An alternative implementation could pass a geometrical shadow mask instead of the texture mask in the stencil buffer. The geometric mask can be created by representing the edges of the shadow contour as found in the shadow edge detection step analytically. However the texture mask ensures that complex shapes can easily be dealt with. For a more complex scene for which texture memory is an issue, a geometric mask can be extracted from the shadow detection and used in the stencil buffer.

The implementation of the shadow volumes method within the larger framework proceeded in collabo-

ration with Cameron Angus.

4.5 Results

The presented method has been tested in two different contexts[5]. One is computer augmented reality where a textured geometric model is used to represent the reality. This model is processed off-line and the shadow generation is displayed on a screen, as is explained in section 4.5.1. The second context is augmented reality which uses a pre-estimated geometry of the real scene and extracts the radiance textures from live video footage, as is explained in section 4.5.2. In both cases the interactions of the virtual objects are computed in real-time. The following two sections discuss the different contexts.

4.5.1 Computer augmented reality

Computer augmented reality (CAR), takes as input a 3D model with radiance textures mapped onto it. The simulated augmented reality is projected onto a non-immersive display device, such as a screen. The main difference with conventional augmented reality is that CAR usually processes the data off-line, while AR simulates all effects at run-time. The following subsections explain the data preparation involved, and take the reader through the 3 different steps of the presented 3-step methodology.

A Data preparation

A 3D model was generated of a real scene containing a desk, a shelf and several books positioned on the shelf, see figure 4.11. The room in which the desk is positioned receives lots of natural daylight through the windows, as can be seen in (a). Therefore, a spotlight was added to illuminate the scene such that distinctive semi-soft shadows are projected onto the desk and shelf. The model is reconstructed using Realviz' ImageModeler 3.5, from a set of 16 LDRIs captured with a Canon 10D EOS [Can]. In (b) a screen shot is given of ImageModeler 3.5 during the modelling phase. The normals (green arrows) and triangle mesh are visible. Images (c) and (d) illustrate the same scene after exporting the model from ImageModeler to VRML. In (c) all triangles are given the same diffuse material, and in (d) the textures are mapped on top of the geometry. The 3D geometric model is a fairly simple model, containing only 96 triangles. No effort was made to find the position of the light source prior to the 3D modelling. Instead the shadow volumes were visualized to find a best match between the real shadows and the estimated real shadows by manually repositioning the virtual light source within the 3D scene.

The model consists of objects with many different materials: the white cloth, the wooden shelf, the glossy books, and the divider covered with blue fabric. For each of these different materials, a shadow factor needs to be calculated. Some surfaces are textured, which in this context means that the material is not homogenous but contains different shades: the shelf shows local BRDF differences, the cloth is wrinkled, and the book covers contain text and images.

B Shadow generation

Figure 4.12 shows the shadow model, obtained by generating the semi-soft shadow mask after shadow detection, see sections 4.4.1 and 4.4.2, and projecting these masks on top of the 3D geometry. The top row images (a) and (b) illustrate the geometric model with the textures mapped onto it. The bottom row illustrates the (inverted[6]) semi-soft shadow masks, also mapped onto the geometric model.

[5] A demo giving an overview of the developed method can be found online at [JAL+04].

[6] This is due to our choice of the implementation of the region growing algorithm that actually grows the shadow as a white region. Prior to using the semi-soft shadow mask during the shadow generation it is inverted.

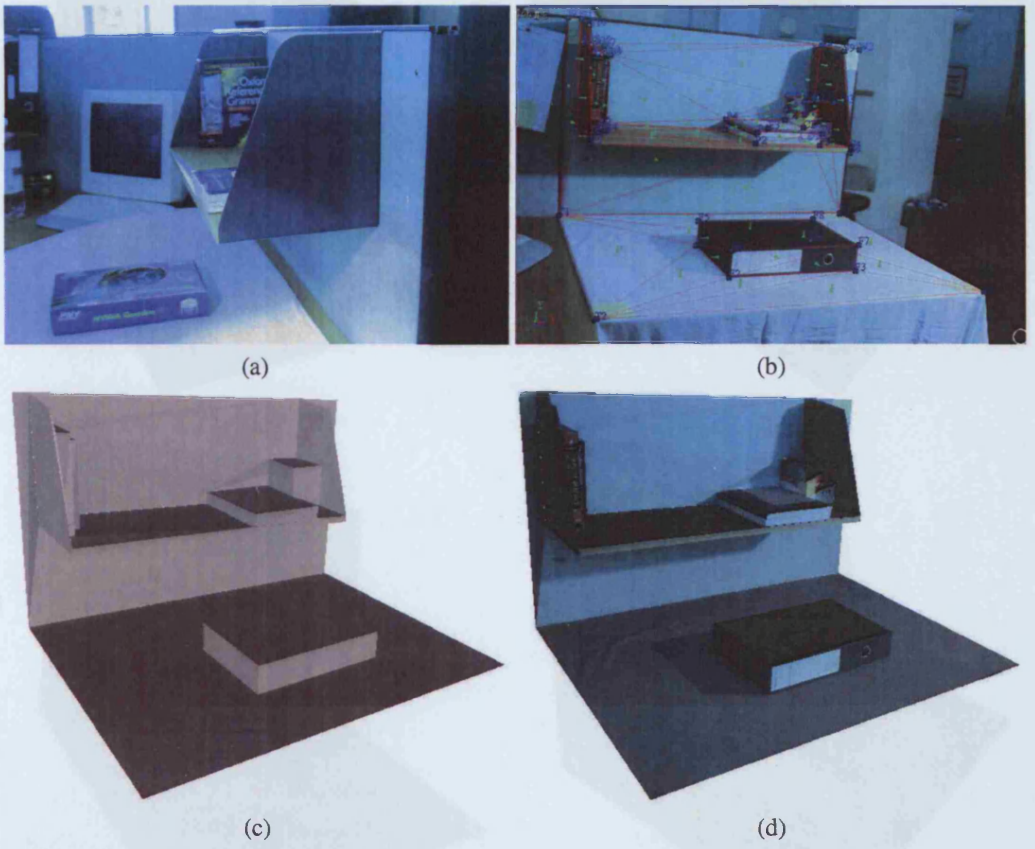


Figure 4.11: *(a) Scene overview: a desk, shelf and several books in a scene receiving lots of daylight coming through the windows. A spotlight was added to the scene to create semi-soft shadows. (b) A screen shot of reconstruction software ImageModeler 3.5. (c) and (d) Screen shots showing the VRML model of the scene. The model consists of 96 triangles (c), onto which textures are mapped (d).*

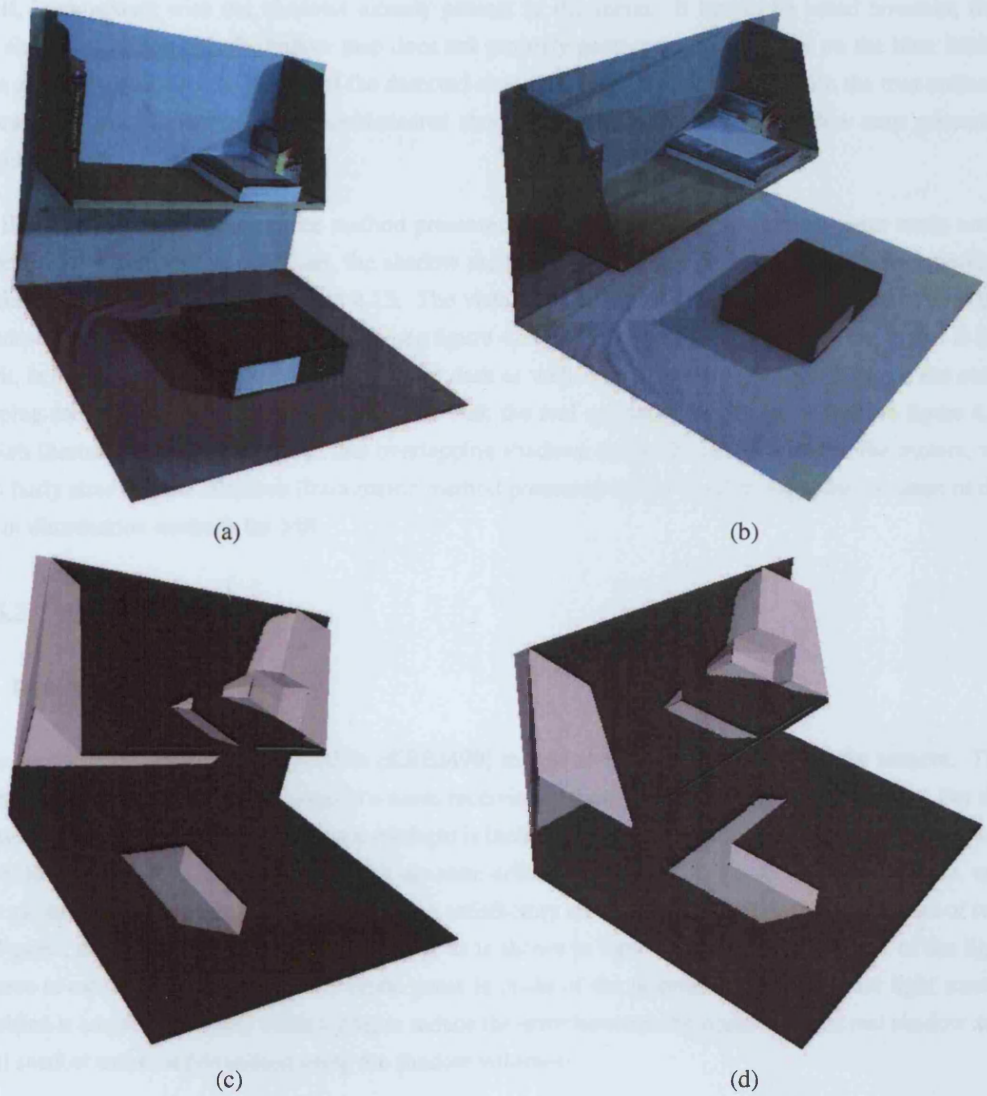


Figure 4.12: *The top row shows the 3D model presented in figure 4.11 from two different viewpoints, the bottom row shows the (inverted) semi-soft shadow masks projected on top of the same 3D geometry from approximately the same viewpoint.*

The 3D scene is augmented with two virtual objects: an avatar, and a box. To simulate the common illumination effects we used shadow volumes to generate the shadows as was discussed in section 4.4.3. While the shadow masks were calculated in advance (off-line), the virtual shadows are simulated at run-time. Figure 4.13 presents the results. The avatar and box move through the scene, the avatar is positioned on the desk, the box floats in the sky and is visible in (a) and (c). In (a), (b), (c) and (d) the avatar casts shadows onto the desk top and blue divider. The box casts shadows onto the shelf and the blue divider. Images (a) and (c) show how the shadow of the box, which is projected onto the wooden shelf, is consistent with the shadows already present in the scene. It has to be noted however, that (d) shows that the semi-soft shadow map does not properly protect the soft shadow on the blue fabric. The gradient applied to the border of the detected shadow region does not match with the true softness character of the shadow. A more sophisticated shadow detection or semi-soft shadow map generator should reduce this type of error.

To illustrate the contribution of the method presented in this chapter and the improvement made compared to the previous state of the art, the shadow method from Haller et al. [HDH03] has been applied to the same scene as shown in figure 4.13. The virtual shadows generated are not consistent with the shadows present in the real scene texture, see figure 4.14. In (a) the virtual shadow of the avatar is too dark, in (b) the virtual shadow of the box is too dark as well. The pixels of the real shadow in the overlapping areas are too dark. If we compare this with the real overlapping shadows shown in figure 4.3, which illustrates that the borders of real overlapping shadows cannot be identified from the texture, we can fairly state that the common illumination method presented in this chapter improves the state of the art in illumination methods for MR.

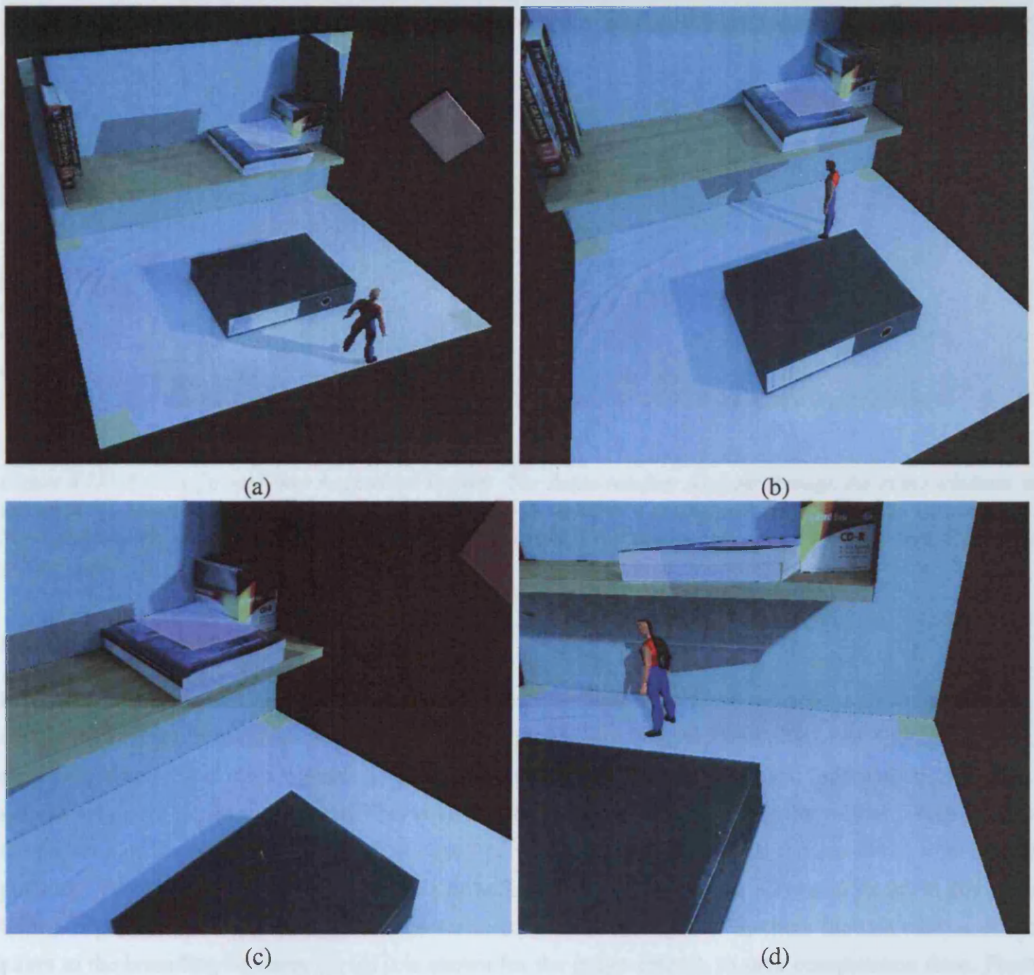
4.5.2 Augmented reality

A Data preparation

The second experiment uses ARToolkit [KBBM99] to perform real-time tracking of the camera. The scene consists of a laptop, positioned in a room receiving natural daylight through the windows. For the same reasons as in the first application a spotlight is included in the 3D scene to ensure clear shadows are cast, see figure 4.15 (a). The shadows cast are semi-soft shadows with only a low level of softness, and the use of the binary shadow mask proved to be satisfactory on this scene. The 3D model consists of two polygons, each forming one face of the laptop, as is shown in figure 4.15 (b). The position of the light source is estimated manually: first a crude guess is made of the position, then the virtual light source position is adapted manually while trying to reduce the error between the position of the real shadow and real shadow estimate (visualised using the shadow volumes).

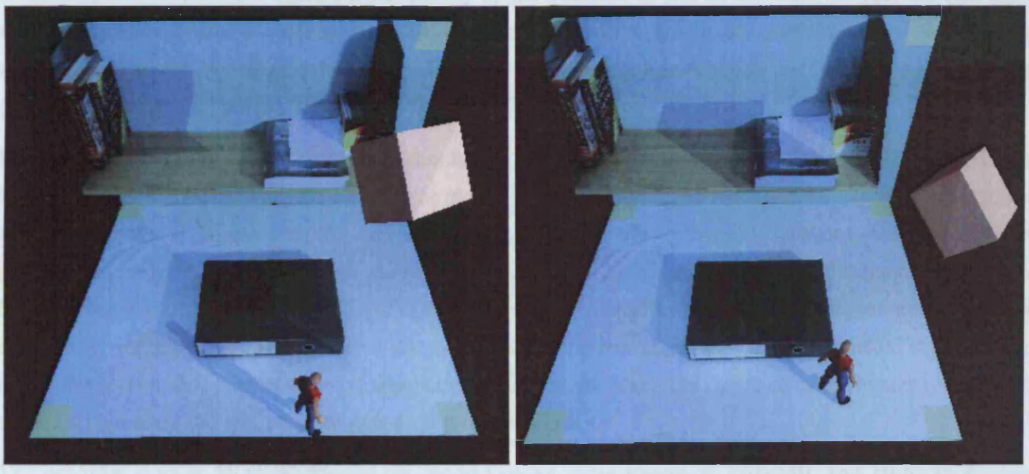
The tracking of the camera using ARToolkit, and the animations added, have been implemented in collaboration with Cameron Angus and Jean-Daniel Nahmias.

The streaming of the video footage used the green colour channel and the shadow factor for the green colour channel. The choice to stream the footage using one colour channel is made for computational reasons. There is no need to perform the shadow detection and shadow protection on the three colour channels simultaneously, therefore only one colour channel is fed into the 3-step mechanism. The visualisation could have been carried out in the three colour channels, as this would simply have required a shadow factor triplet, which would not have increased the overall computation time to generate the shadows.



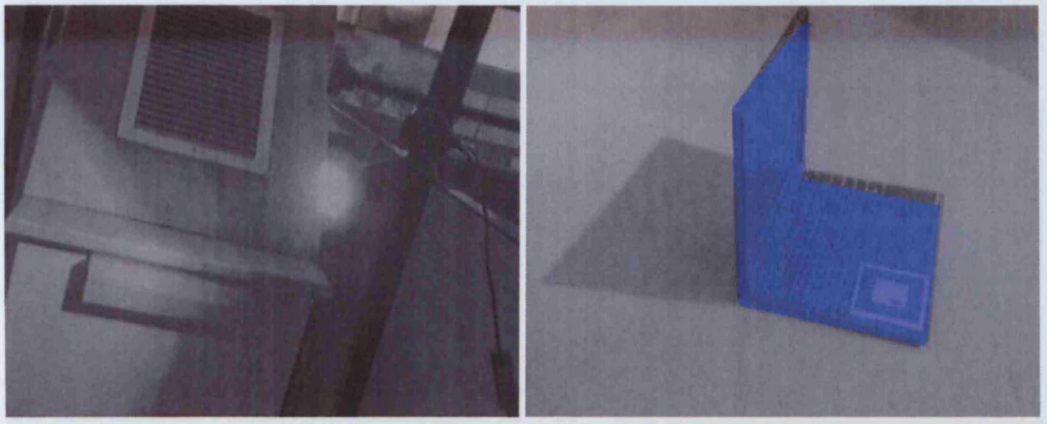
(a) (b) (c) (d)

Figure 4.13: *Results for Computer Augmented Reality: the virtual shadows are consistent with the real shadows already present in the scene texture.*



(a) (b)

Figure 4.14: *For benchmarking purpose the method from Haller et al. [HDH03] has been applied to the same scene. The virtual shadows are inconsistent with the real shadows present in the scene texture. The pixels inside a real and virtual shadow are too dark.*



(a) (b)

Figure 4.15: *Results for real-time Augmented Reality. The scene receives daylight through the many windows in the room. To ensure the shadows are clearly defined on the ground, a simple spotlight is added to the scene (a). The laptop model (overlayed in blue) is very simple, consisting of just two planes, one for the top panel, one for the ground panel (b).*

B Shadow generation

It was already discussed in section 4.2 that ARToolkit suffers from tracking errors, resulting in an incorrect camera position retrieval. Furthermore this inaccuracy is inconsistent between different frames, especially when the camera moves. Together with the approximated geometric reconstruction, this results in very poor shadow estimates. This is visualized in figure 4.16 (a) where the yellow pixels indicate the position of the estimated real shadow based on the geometric model and the estimated light source position. From the real shadow estimate, a bounding box is constructed, shown in green in (b). The Canny edge detector finds all edges in the scene (c), the bounding box restricts the calculation of the edges to the bounding box area (in (c) it is shown for the entire frame), to save computation time. From the centre of the bounding box a seed pixel is grown into the shadow region shown in yellow in (d), using a straightforward region growing algorithm.

To illustrate the robustness against the accuracy of the position of the virtual light source, the real light source was repositioned to various different locations at run-time, the position of the virtual light source remains unchanged. As long as the centre of the green bounding box remains inside the real shadow, the estimate is accurate. This is illustrated in figure 4.17.

In figure 4.18 examples of the shadow generation are shown with a virtual avatar walking around a laptop. In (a) the avatar's shadow does not overlap with the laptop's real shadow. In (b) the avatar's shadow partially overlaps with the real shadow, the shadow mask correctly prevents the real shadow pixels from being scaled along with the other pixels inside the virtual shadow. This image also shows how the avatar is occluded from the light source as well: the shadow cast by the real object onto the avatar is correctly simulated. Image (c) shows the avatar inside the real shadow. The entire virtual shadow falls inside the real shadow, therefore no scaling takes place. In addition, the radiance of the avatar is correctly scaled, to simulate the real shadow cast on the avatar's body.

The program runs at 15 to 30 frames per second on an input image of 320x240, and 10 to 15 frames per second on an input image of 640x480. These timings were recorded on a HP 3000+ Athlon with a GeForce FX 5950 Ultra. These timings include all computations and the camera capture at each frame; except for the geometry extraction, the light source registration, and the shadow factor calculations no pre-computations are made. The difference in frame rate is due to the shadow calculations. For instance,

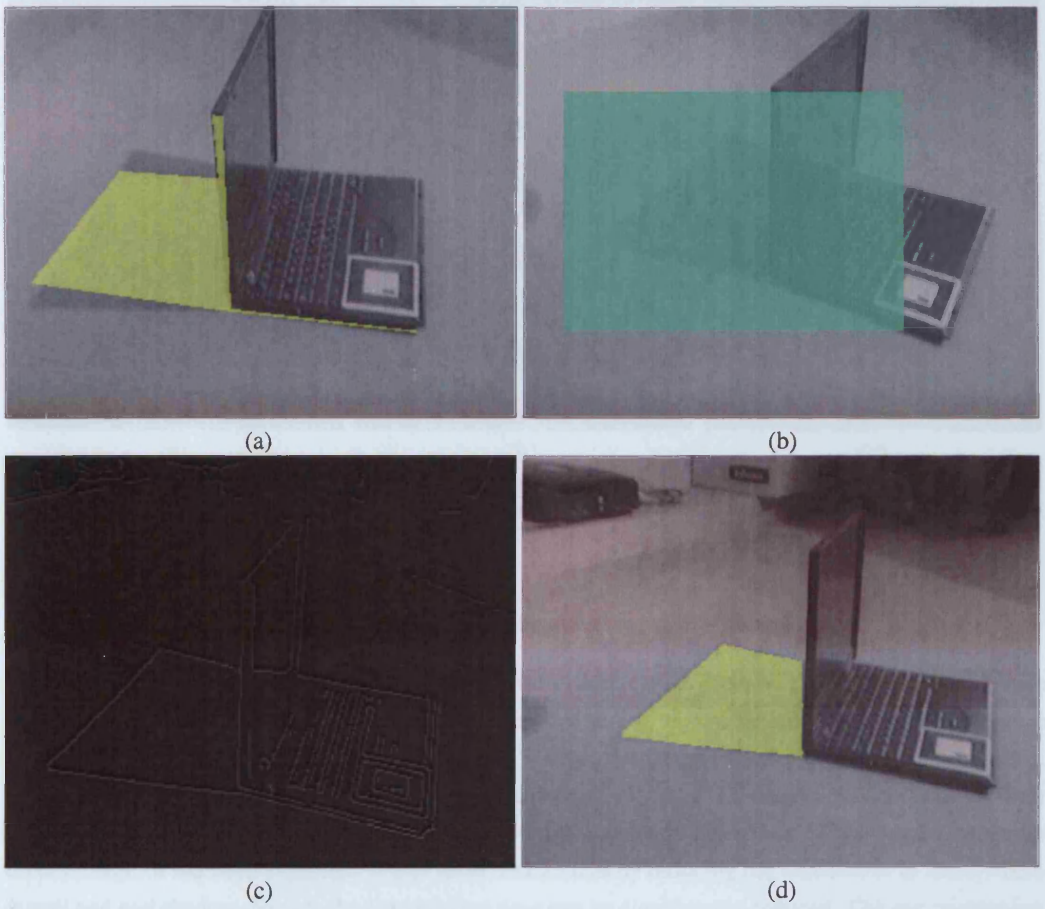
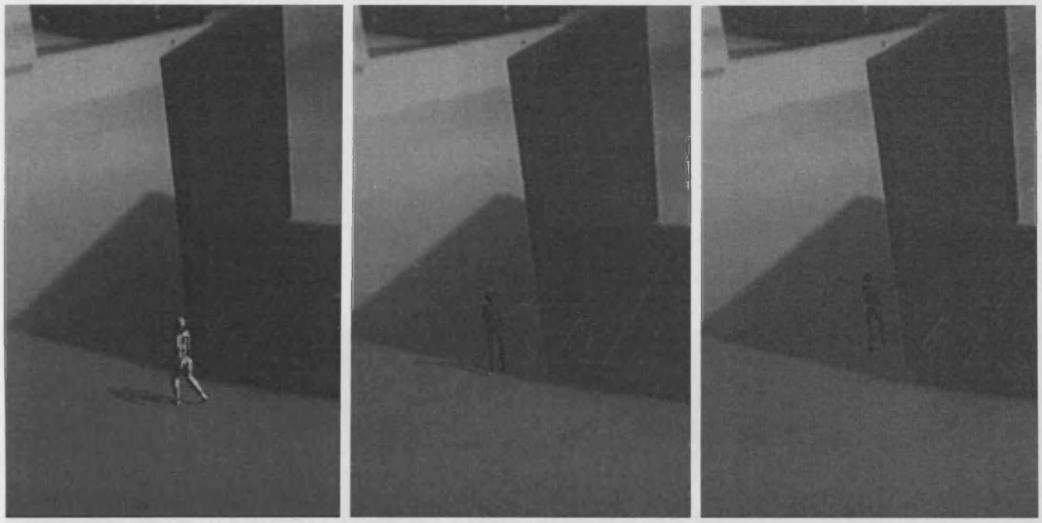


Figure 4.16: *(a) The real shadow estimate is shown in yellow. There is a clear mismatch with the real shadow. (b) Search region extracted from the shadow in (a), in which the shadow region detection is performed. (c) Result of the edge detector. (d) Result of the shadow mask generation.*

Figure 4.17: *Robustness against the inaccuracies in light source position estimation: the light source was moved to different locations at run-time, the position of the shadow was accurately detected.*



(a) (b) (c)

Figure 4.18: *Results of the algorithm for an animated virtual avatar walking around a real laptop. Shadows are automatically detected and generated using our real-time algorithm.*

the larger the real shadow, or the higher the resolution, the slower the region growing.

The method works in real-time for the examples shown in this section. However, it is important to discuss the scalability of the method to more complex scenes, containing more polygons. The total computation time depends on the time to calculate the shadow border, the time to grow the real shadow estimate from a seed pixel and the time to compute the virtual shadows using the shadow volumes. It has been explained in section 4.4.1 that the shadow detection time is mostly dependent on the resolution of the input footage. It has been explained in that same section that by reducing the search area to areas where virtual and real shadow overlap, the computation time can be significantly reduced. The region growing time is similarly dependent on the real shadow size, and will increase proportionally to the size of the real shadow. During the experiments it became clear that the main contributor to the computation time is the region growing. This is understandable as the region growing is a recursive process. The virtual shadow generation depends on the complexity of the virtual object, the more polygons, the longer it will take to render the virtual shadow. Nevertheless, shadow generation using shadow volumes can be considered to be a real-time process.

Figure 4.19 illustrates the limitations of the presented method. Image (a) illustrates the bounding box, which is generated from the real shadow estimate. The region growing starts from the centre of this box. The shadow mask is shown in yellow in (b), the shadow is generated through region growing inside the shadow contour. Due to noise, certain noisy edge pixels can exist, which can hamper the region growing. For instance the top white box shown in (b) illustrates a small part of the shadow region that is not detected, while in the bottom white box the alignment is sufficient. After re-positioning the real light source in (c), the green bounding box is totally misplaced compared to the position of the real shadow. As a result, the start seed of the region growing does not lie in the centre of the real shadow. Due to noisy edge pixels, the region growing fails to deliver the entire shadow map, instead a small part of the true shadow is returned. Image (d) shows further misalignments between the real shadow contour and the shadow contour estimate in the white boxes. The softer the shadow contour, the more likely such misalignments occur. It should be noted that in general the misalignments remained unnoticeable, and that it only occurred for a few frames in the video footage.

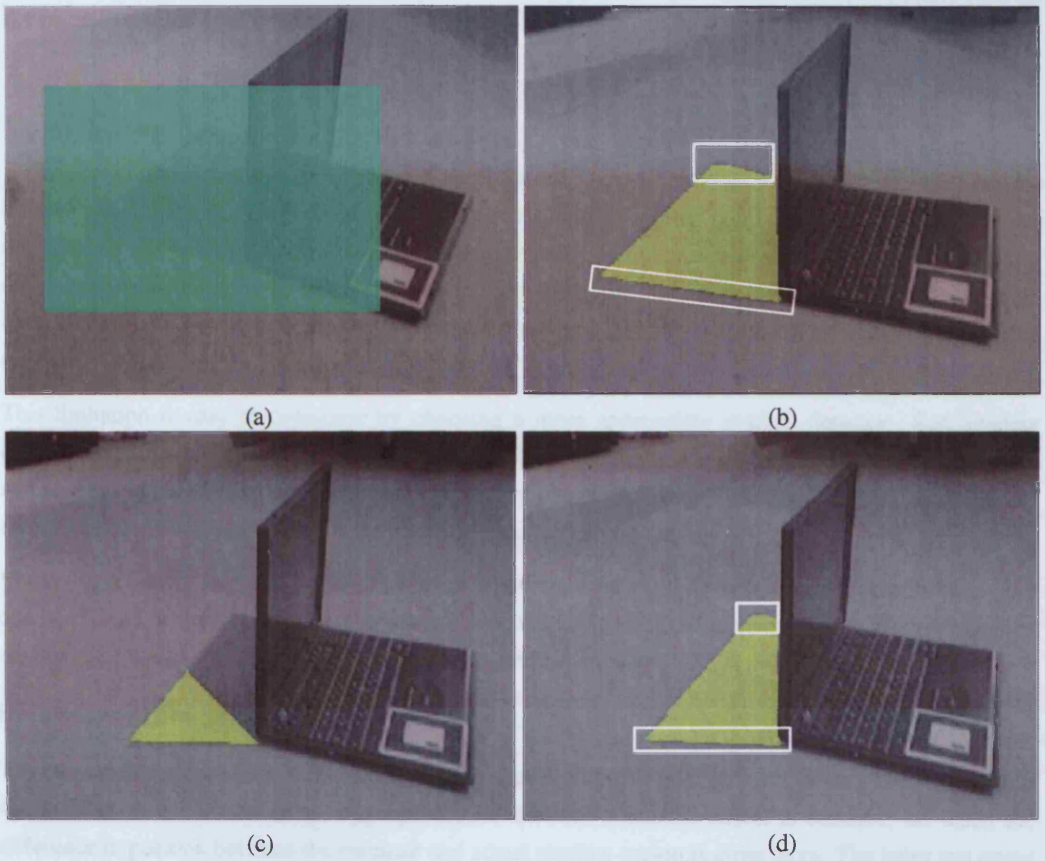


Figure 4.19: *These images illustrate the limits of the shadow generation. (a) The shadow boundary box is given by the green box, this bounding box is defined by the reconstructed real shadow which in turn is defined by the reconstruction model and the estimated light source position. (b) The shadow mask shown in yellow is derived from the shadow contour. The shadow contour is a one-pixel thick contour. In the top white box, we can see that this contour might not be completely aligned with the actual shadow contour. The alignment is good in the bottom white box. (c) When noise is present after the edge detection, this might tamper with the region growing, which result an incorrect (incomplete in the example shown) shadow mask. (d) Another example of misalignment between shadow contour and actual shadow contour is shown in the white boxes.*

4.6 Limitations and possible improvements

The 3-step methodology presented in this chapter generates common illumination for augmented reality. More precisely, the method enables the generation of virtual shadows consistent with shadows already present in the real environment. The strength of the presented method is that it succeeds in simulating consistent shadows in a low-cost manner, no special equipment is required, and in real-time. With most calculations occurring in real-time, the developed method is user-friendly. It was already mentioned that the presented algorithm allows improvement, this section lists the current limitations of the implementation of the methodology as presented in section 4.4 and 4.5, and provides a selection of improvements the implementation could benefit from.

A Soft shadows

The focus of this thesis and this chapter is on finding problems with and improvements for illumination methods. To illustrate the consequences of an incorrect geometric model a semi-soft shadow detector has been implemented, which suffices to illustrate our conclusion that poorly reconstructed models will create artefacts when simulating shadows through scaling. The characteristics of the Canny edge detector only allow *strong* edges to be detected. By changing the upper and lower thresholds of the Canny edge detector, the user can manipulate the strength of the edges that are detectable. However, lowering the strength will automatically increase the detection of *false* edges.

This limitation is easy to overcome by choosing a more appropriate shadow detector. Soft shadow detection is more difficult to achieve than hard shadow detection, mainly due to the lack of features in a soft shadow which can ease the detection process. For this particular application, however, a soft-shadow detection could make use of the geometrical estimate of the real shadow.

Improving the shadow detection would also enable the use of multiple light sources, compared to only one as is used in the current implementation. As was indicated in this chapter, soft shadow maps or environment maps could be used to simulate the virtual shadows.

B Shadow region growing

The current region growing algorithm, picks as a start seed the middle of the bounding box, see section 4.4.2. This choice is not always appropriate, for instance when the object is concave, or, when the difference in position between the estimate and actual shadow region is quite large. The latter can occur when the geometry is known very poorly, or when the virtual light source is incorrectly positioned.

Finding the seed pixel from which to grow the shadow region could be improved by using the knowledge of the shape of the shadow contour. The characteristics of the shadow intensity can be used as well. In most cases the darkest pixels in the bounding box would belong to a real shadow, although this cannot be generalized and this assumption can result in errors.

C Shadow factor

In the current implementation, the shadow factor is defined per material, by finding the scaling factor for the intensity between of a group of pixels inside and outside (but in the near vicinity of) the shadow region. For different materials, different shadow factors are defined.

However, the shadow intensity does not only depend on the type of material, but also on the orientation of the surface towards the light source and the orientation to the observer, especially for specular or glossy materials. Therefore a shadow factor defined on a per-pixel basis will most likely improve the quality of the shadow generation. In particular when used in an environment casting soft shadows.

In our implementation the shadow factors are calculated manually. It has been indicated in this chapter that it is relatively straightforward to calculate the scaling factor automatically. This automatic implementation has been left as future work, as a more sophisticated shadow factor calculation is required anyway.

D Light Source tracking

The robustness of the current implementation against misplacement of the virtual light source was already discussed in section 4.5.2, see figure 4.17. This indicates that even a crude estimate for the light source position can be used to detect the real shadows. Once the correct shadow region is found, the deviation from the estimated shadow position can be used to fine-tune the virtual light source position. Also, changes in the real shadow region, due to light movement, can be used to track the light source in the real scene.

4.7 Chapter summary

In this chapter, problems associated with inaccurate geometric reconstructions for common illumination solutions were highlighted. In order to produce realistic virtual shadows, the virtual shadows need to be consistent with the real shadows that might already be present in the real environment. Consistent means that the virtual shadows need to be simulated into the correct direction, and with the correct intensities or colours. For real-time augmented reality, shadow volumes can be used to generate the virtual shadows. The pixels inside a virtual shadow are simply scaled to simulate the blocking of light due to the virtual object.

Problems arise when virtual and real shadows overlap. The scaling applied to the pixels inside a real shadow needs to be different from the scaling applied to the pixels outside a real shadow. When the scene consists of mainly hard shadows, the scaling of pixels inside a real shadow should in fact not take place at all, as this would in principle imply a double scaling (once due to the real shadow, and once due to the virtual shadow). This problem could in fact be reduced by calculating the position of the real shadow based on the scene's geometric model and the position of the light source. However, this real shadow estimate usually does not entirely overlap with the real shadow visible in the scene (or in the texture), because it is very sensitive to the estimated light source position and the geometry of the scene objects.

A solution would be to focus our efforts on improving the reconstruction of a 3D scene. It has been mentioned in this chapter that using a 3D scanner does not necessarily solve the problems of user-aided reconstruction software. Each of the two reconstruction tools has its own flaws, and relying on an accurate geometric model for AR, limits the applicability of the AR methods to well-defined, controlled environments. Therefore a better solution is to tackle the problem of generating consistent shadows for inaccurate geometric models.

This is achieved through detecting the actual shadow region inside the radiance texture or input image using image processing. This shadow region is then used to block the scaling of the pixels inside a virtual shadow region. The proposed methodology for shadow generation consists of three steps: shadow detection, shadow protection and shadow generation. The implementation of this 3-step algorithm focuses on scenes with semi-soft shadows through the generation of a semi-soft shadow mask. The scaling factors used to generate the virtual shadows are calculated per material present in the scene.

The method has been applied to two different contexts; in both contexts the method proved to be successful. A first is computer augmented reality, the second plain real-time AR. For both applications,

the resulting virtual shadows are consistent with the real shadows already present in the scene. The AR application illustrated the real-time character of the implementation: the shadows are detected and generated at run-time and in real-time. This allows at run-time user navigation of the virtual and real objects and even the light source.

The limitations for the presented method lie in the assumptions made: only one main light sources is present in the scene to cast semi-soft shadows. An improvement to the current implementation would include an upgrade to a consistent soft shadow detector and generator. A further improvement would be to allow the system to refine the positions of the light sources in the scene using automatic light tracking.

The presented methodology provides common illumination, nevertheless the framework could also be used to improve the robustness of relighting methods against an inaccurate geometric reconstruction. Every time a lighting effect needs to be identified or removed, it is important to have an accurate geometric model. If this model is inaccurate, the lighting effect needs to be identified using a different means, which in the context of shadow generation is a shadow detector as used in this chapter.

Chapter 5

HDRI generation for dynamic scenes

5.1 Introduction

Application domains such as image-based rendering and mixed reality use photogrammetry when performing relighting and require input directly from photographs, as is clarified in chapters 2 and 3. Photographs often present a loss of colour information since clipping and noise occur in areas which are under- or over-exposed. This loss of information can have a crucial impact on the accuracy of the photogrammetric results. Methods have been created to combine information acquired from conventional or *low dynamic range images (LDRIs)* captured with varying exposure settings, creating a new photograph with a higher range of colour information called a *high dynamic range image (HDRI)*. It has now become viable to use HDRIs in photogrammetry, and new cameras [BAS][NB03][NBB04] are being developed with a higher dynamic range than the conventional cameras.

The drawback of generating an HDRI from a set of LDRIs is that the total capture time with a standard camera is at least the sum of the exposure times used for programmable cameras. It can increase further for non-programmable cameras as the user needs to change the exposure setting manually between the captures. However, between each LDRI capture, the environment can change or the camera can move. This is especially true for uncontrollable, outdoor scenes and when not using a tripod. In such cases, combining LDRIs results in an incorrect irradiance reconstruction when using the currently available HDRI generation tools, such as HDRshop [HDR], Rascal [Ras][MN99] or Photomatix [Pho].

In this chapter a new framework is presented for HDRI generation. This framework takes a set of LDRIs as input and produces an HDRI. The set of LDRIs can be captured with a hand-held camera, and some object movement is allowed in the scene. The produced HDRI is free from any artefacts that standard HDRI generation methods would produce due to the movements. In particular, the method presented improves and complements the limited alignment and object movement removal used by Photosphere [Any].

This chapter is organized as follows. Section 5.2 illustrates theoretically the flaws that exist in an HDRI due to object movement, and defines the scope of this chapter. From the explanation given it should be clear why camera and object movement are avoided throughout the LDR capture, and what types of errors we can expect when these conditions are not met. An overview of the current advances made for camera and object movement removal in the context of HDR imaging is presented in section 5.3. Section 5.4 presents the methodology developed in this thesis to compensate for camera and object movement during the HDRI capture. The methodology contains a movement removal step, which consists of two separate processes. Firstly, the camera movement removal, described in section 5.5, and then object movement

removal, described in section 5.6. Section 5.7 analyses the presented method based on several examples and section 5.8 discusses its limitations. Finally, a summary and conclusion are given in section 5.9.

Part of the work presented in this chapter has been presented at Siggraph 2006 as a Technical Sketch [JWL05] and has been completed in collaboration with Greg Ward from Anyhere Software, and prime developer of Photosphere [Any]. The work presented in this chapter is my own work, except for those parts that have been appropriately referenced. Several of the exposures used during the HDRI generation testing phase have been captured by Greg Ward and Celine Loscos.

5.2 Dynamic capture

Chapter 2 provided an overview of HDR imaging, and section 2.5.2 discussed the method commonly used to generate HDRIs from different exposures. According to the described methodology we can conclude that corresponding pixels in the sequence of exposures should have the same irradiance values after the pixel transformation from intensities to irradiance values, if saturation and under-exposure effects are ignored. This implies that the following conditions need to be met:

- During the exposure capture, the scene illumination must be static. If the illumination changes, corresponding pixels between different LDRIs represent a different irradiance content. The exposure capture time can vary between several seconds to a minute, depending on the need of the exposure settings to be set manually.
- The camera position and viewing direction needs to be static. This can be achieved by using a tripod or by positioning the camera on a static ground plane. If the camera moves the pixels between the different LDRIs are misaligned and represent different scene content.
- All objects need to be static. If an object moves throughout the image capture, the pixels in the area affected by its movement show different scene content. The longer the capture time, the more likely a scene object will move or deform, especially in uncontrollable environments.

If the camera curve is generated in advance, prior to the exposure capture, the intensity values in the LDRIs can be transformed to irradiance values. Ignoring quantization and image noise, the resulting irradiance value of a pixel should be equal to the actual irradiance value of the scene point represented by that pixel. Irradiance values deviate from this expected value either due to saturation effects, or due to failing to meet any of the conditions mentioned above. If the deviation is due to object and camera movement, the irradiance values can be recovered as is discussed in sections 5.5 and 5.6, even if some pixels are saturated. However, if the deviation is due to illumination changes, it is difficult, if not impossible, to reconstruct the irradiance values. One might decide to discard an exposure showing illumination changes from the total set, but there might be too little irradiance information from the other exposures. If the camera curve is unknown a priori, identifying the illumination changes will be difficult without the use of an external device such as a lightmeter. Dealing with illumination changes throughout the exposure sequence has been left as future work.

Figure 5.1 gives a schematic overview of the implications of object movement during the LDRI capture using a synthetic example. The figure follows the same methodology as shown in figure 2.6. In the final HDRI, the moving object appears blurred, or smeared out. This is often referred to as a *ghosting effect.* In the example shown, the pentagon gives the impression of being present three times in the final HDRI, due to the merging of the exposures into one HDRI. Camera movement would result in a similar illusion, most likely even more disastrous as it would apply a ghosting effect on all objects in the scene.

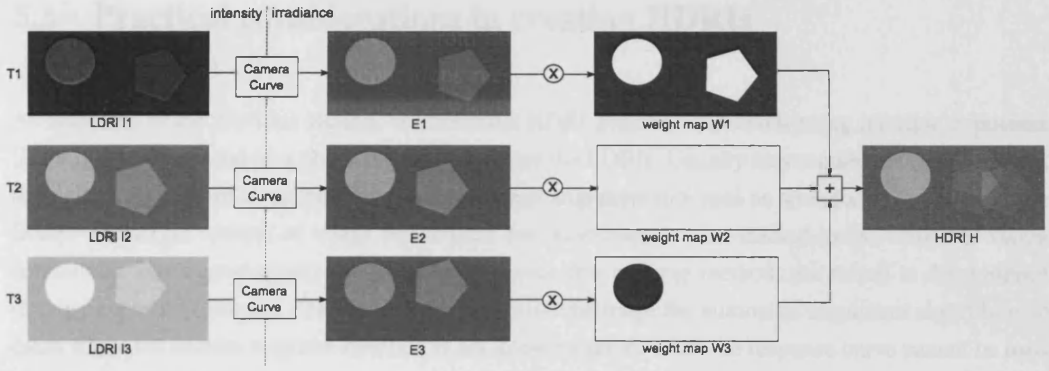


Figure 5.1: *The pentagon moves between the three different exposures shown, and does not show saturation or under-exposure in any of the I_is. In the HDRI, generated as a weighted average of the I_is, the pentagon appears three times, with a different strength. This is called a ghosting effect.*

To avoid camera movements, the camera should be positioned on a tripod. However, from experience, we have learned that even with a tripod small camera movements cannot always be prevented. When a heavy tele-lens is used to capture, for instance, a reflective sphere, the lens can move slightly due to the gravity and an unstable tripod. Furthermore, if the camera settings are changed manually, the manual manipulations can cause the tripod to move.

In this thesis the current state of the art in image alignment for HDRI generation is improved. The alignment algorithm developed focuses on a sequence of exposures captured with a hand-held camera. Such an alignment method enables HDRI capture at remote sites, without the extra cost of carrying a tripod. We also developed a novel movement removal method, which removes the errors due to object movement in the final HDRI. This enables the HDR capture of sites with people, cars and clouds, and scenes subject to windy conditions. The combination of the alignment algorithm and the movement removal method allows HDR reconstruction of images of real-world scenes, such as outdoor scenery and public spaces.

Before actually discussing the developed method, we would like to make three assumptions:

- The camera movement is limited to an euclidean transformation, rather than a perspective transformation. It is fair to assume that as long as the photographer tries to keep the movement to a minimum, by holding the camera as static as possible, while pointing the camera into the same direction, the misalignments between the different LDRIs are small. Usually the hand movements can be approximated by a rotation around the viewing direction and a translation up and down and from left to right, but perpendicular to the viewing direction.

- The movement removal process makes the assumption that each moving object should be visible without saturation in at least one exposure. For the objects considered during the experiments in this chapter, this was usually the case.

- The illumination conditions are assumed to have not changed during the capture of the set of exposures. This means that the illumination should not change during several seconds for programmable cameras and for about one minutes for non-programmable cameras.

5.3 Practical considerations in creating HDRIs

As discussed in the previous section, conventional HDRI generation methods using multiple exposures [MP95][DM97] depend on a good alignment between the LDRIs. Usually they require the use of a tripod throughout the capture, some provide a manual image alignment tool such as provided in the Rascal suite [Ras]. The larger context of image registration and alignment is well-studied in the computer vision community. For a good survey see [Bro92]. However, few of these methods are robust in the presence of large exposure changes. This presents a particular challenge for automatic alignment algorithms in cases where the camera response function is not known a priori, since the response curve cannot be used to normalize the LDRIs in a way that would make feature detection and matching reliable.

Three solutions have been presented for image alignment in an HDRI building context. Ward [War04] introduced the median threshold bitmap (MTB) procedure, which is insensitive to camera response and exposure changes, demonstrating robust translational alignment. Bitmap methods such as MTB are fast, but ill-suited to generate the dense optical flow fields employed in local image registration. A limited alignment procedure using the MTB procedure has been implemented in the software package Photosphere [Any]. Kang et al. [KUWS03] presented a method that relies on the camera response function to normalize the LDRIs and perform local image alignment using gradient-based optical flow. Sand and Teller [ST04a] presented a feature-based method, which incorporates a local contrast and brightness normalization method that does not require knowledge of the camera response curve [ST04b]. Their match generation method is robust to changes in exposure and illumination, but faces challenges when few high-contrast features are available, or features are so dense that matches become erratic. This is often the case for natural scenes, whose moving water, clouds, flora and fauna provide few static features to establish even a low-resolution motion field. This is where both papers bring in sophisticated techniques, hierarchical homography in the case of Kang et al., and locally weighted regression in the case of Sand and Teller, to overcome uncertainties in the image flow field. Even so, local image warping becomes less reliable as contrast decreases, leading to loss of detail in regions of the image. Furthermore, moving objects may obscure parts of the scene in some exposures and reveal them in others, This leads to the optical flow parallax problem, where there is not enough information at the right exposure to reconstruct plausible HDR values over the entire image.

The main reason for warping pixels locally between the LDRIs is to avoid blurring and ghosting in the HDRI composite. With the presented method in this chapter, the need for image warping is removed by observing that each LDRI is a self-consistent snapshot in time, and in regions where blending images would cause blurring or ghosting due to local motions, an appropriate choice of input LDRI to represent the motion is sufficient. This approach allows us to apply robust statistics for determining where and when blending is inadequate, and avoids the need for parallax fill. Certain regions may be slightly noisier than they would be with a full blend, but this is an accustomed form of image degradation, and preferable to the ghosting effects that result from improper warping and parallax errors.

Chapter 2 mentioned the development of HDR cameras, and even HDR video. However, the problem of non-static environments remains even for these tools to capture HDRIs. With HDR cameras, the time required to take a picture decreases but always remains greater than the longest exposure time used to capture the set of LDRIs. Many of the methods we describe could also be incorporated in HDRI cameras, to reduce the appearance of artefacts.

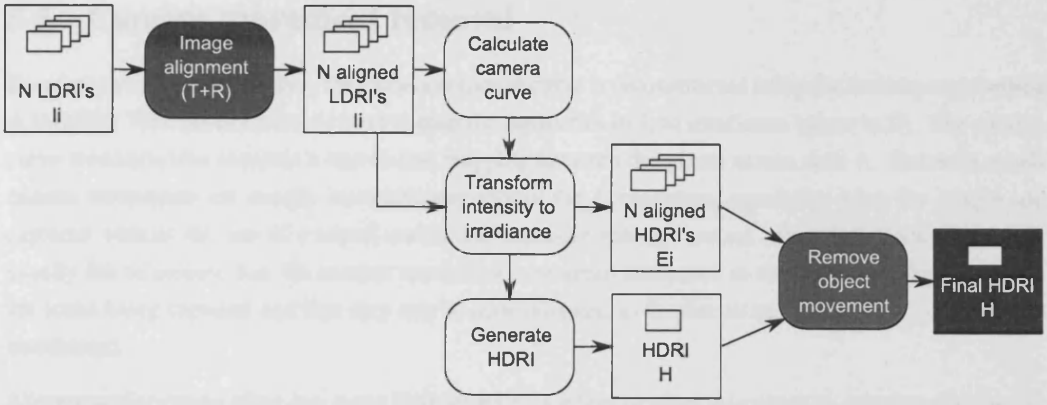


Figure 5.2: *HDRI generation methodology: the white and grey rounded boxes are processes that operate on input data and produce output data, both indicated by the white squares. The grey rounded boxes are modules developed in this thesis.*

5.4 Methodology

An overview of the HDRI generation methodology is presented in figure 5.2. The rounded boxes represent processes that operate on the input data and produce output data, both illustrated by square boxes. To compensate for the camera and object movement, two extra modules have been developed in this thesis, that fit into the current HDRI generation methodology. These two extra modules are represented by grey-coloured rounded boxes. The processes involved in the HDRI generation are:

- **Image alignment**: to calculate the camera curve and to combine the LDRIs I_i into an HDRI H, the scene content in the images needs to be aligned. The camera misalignments are approximated by a rotation around and a translation perpendicular to the viewing direction. The alignment method is designed for the alignment of different exposures, which have certain specific characteristics that make conventional alignment methods unpractical. This is explained in section 5.5.

- **Camera curve calculation**: the camera is calibrated from a set of LDRIs. The camera curve defines the relation between scene irradiance and image intensity, for a certain exposure setting. The camera curve is usually represented as an N^{th} degree polynomial.

- **Intensity to irradiance transformation**: the intensity values in each LDRI I_i are transformed to irradiance values, resulting in irradiance images E_i. The transformation is defined by the camera curve and the exposure settings for each LDRI. The transformation is necessary, since only then the LDRIs can be combined into a meaningful HDRI.

- **HDRI generation**: the irradiance values in the images E_i are combined into an HDRI, as a weighted average, according to equation 2.32. The weights are defined per exposure and per pixel and are normalized for each pixel, across the different LDRIs. They should remove the influence of saturated or under-exposed pixels in the LDRIs on the final HDR pixel values.

- **Object movement removal**: non-rigid and moving objects introduce a ghosting effect in the final HDRI. This ghosting effect is removed from the final HDRI in a post-processing step. As is discussed in section 5.6, two different strategies have been developed to remove two different kinds of object movement that can occur in a real-world environment. The movement removal is preceded by a movement detection phase. Please note that the overview presented in figure 5.2 makes abstraction of how and when the movement detection occurs within this framework.

5.5 Camera movement removal

Based on the set of N LDRIs I_i the camera response curve is reconstructed using the technique presented in [MN99]. This camera curve is used to map the intensities in I_i to irradiance values in E_i. The camera curve reconstruction assumes a one-to-one mapping between the pixels across each I_i. However, small camera movements are usually inevitable throughout the I_i capturing, especially when the images are captured without the use of a tripod and/or the exposure settings are set manually. Fortunately, it is usually fair to assume that the camera movements are small compared to the geometric dimensions of the scene being captured and that they can be approximated as Euclidean transformations (rotation and translation).

Alignment algorithms often use scene features such as edges or pixel intensities to calculate the camera transformations. Detecting similar scene features in the I_i's is error-prone as they often represent different scene contents: different intensities, different colours and edges due to under- or over-exposure effects. An example is shown in figure 5.3, which shows two exposures (a) and (b) with (a) showing saturation (several parts of the image are saturated to white). Two edge maps, resulting from applying an edge detector on (a) and (b), are shown in respectively (c) and (d). The same Canny Edge detector has been used as presented in chapter 4. Different thresholds are used for each of the exposures. The thresholds were chosen to improve the edge detection in each of the exposures. The upper and lower thresholds used are $[0.1, 0.3]$ and $[0.3, 0.7]$ for (c) and (d) respectively. However, (c) and (d) show that the edge detector fails to detect the same edges in both images. The difference in the detected shadow edges (especially of the bench on the ground) might contribute to an incorrect alignment.

To align the I_i's effectively, the median threshold bitmap (MTB) technique [War04] is adopted, which uses the median intensity value (MIV) of an exposure I_i as a threshold to transform I_i into a binary image $\widehat{I}_i$. MIV splits the pixels in the images I_i into approximately the same two groups, when saturation effects are kept to a minimum. The reason is that the histograms of the different exposures are similar up to an affine transformation (skewing, translation, but preserving order). In other words, if the value of pixel t in L_1 is smaller than the value of pixel s in L_1, then the value of pixel t in L_2 is also smaller than the pixel value of s in L_2, unless pixel t and s are saturated in L_2 or represent a different scene point due to object movement. An example of two binary images obtained with the MTB technique is given in figure 5.3 (e) and (f). This time, similar scene features have been detected, which makes alignment more robust.

The alignment will make use of the MTB technique to align two exposures. The XOR difference between two binary images $\widehat{I}_i$ and $\widehat{I}_j$ obtained after applying the MTB transform on two exposures I_i and I_j gives a measure for error. The alignment procedure will find the best transformation $T(\cdot)$, consisting of a translation vector $[T_x, T_y]$ and rotation angle α, that when applied to I_i results in the maximum correlation between the two binary images $\widehat{T(I_i)}$ and $\widehat{I}_j$. The alignment described in [War04] reduces this transformation $T(\cdot)$ to a translation while in this chapter the rotation around the centre of the image is considered as well. The search for the best transformation $T(\cdot)$ is similar to that described in [War04], a brief overview is given here, but more details can be found in the previous mentioned paper.

Finding the best translation involves finding the vector $[T_x, T_y]$ that can be applied to the pixel coordinates (k, l). Finding the best rotation involves finding the rotation angle α that rotates the image I_i (hence its coordinates) around its centre. Translating image I_i with dimension $P \times Q$ involves shifting the image. Rotating the pixel values in an image requires 4 multiplications per pixel using a 2D rotation matrix, and a bi-linear interpolation. Therefore finding the optimal transformation is heavily dependent on the resolution of an image.

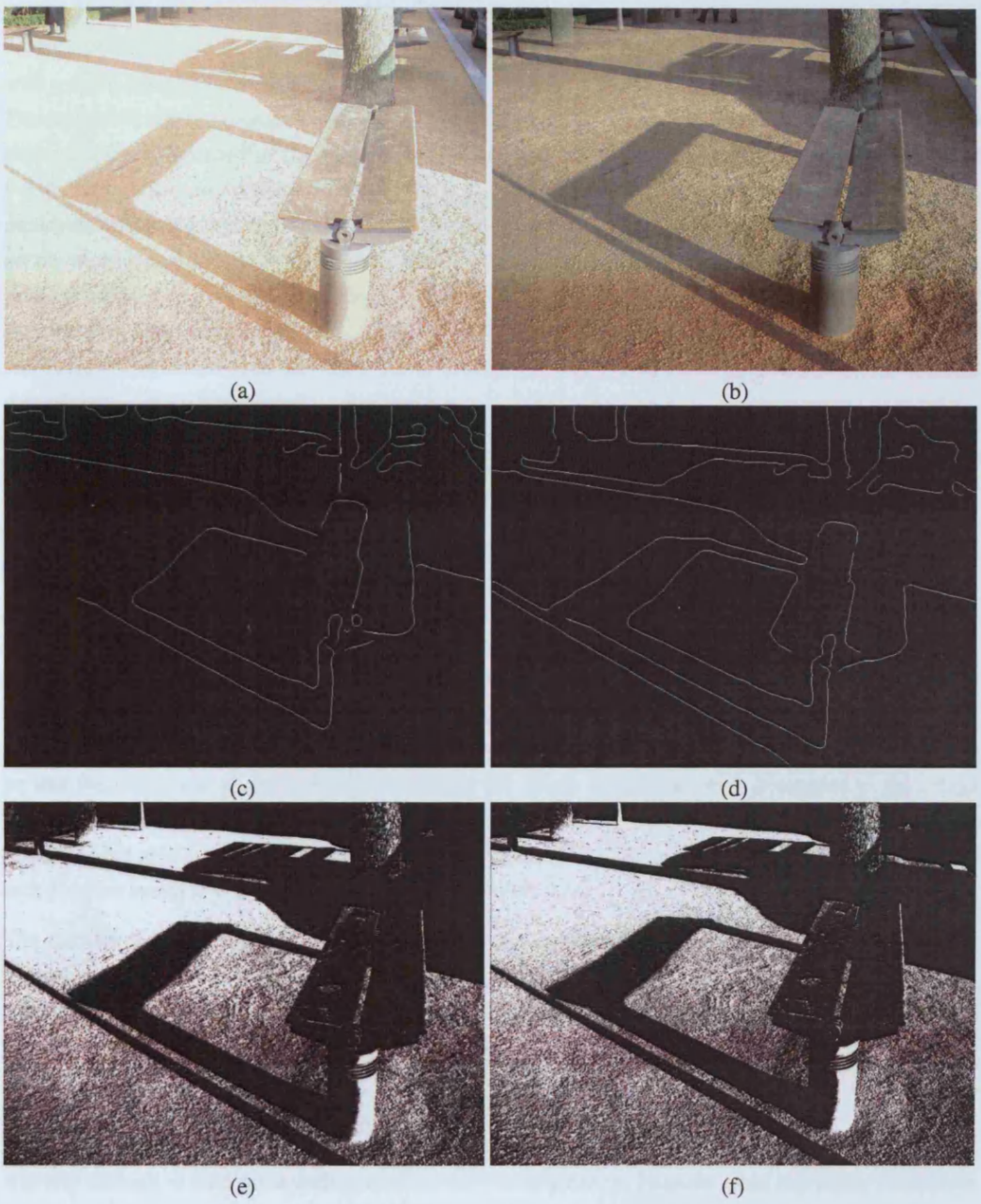


Figure 5.3: *(a)(b) Two different LDRIs captured with different exposures settings. (c)(d) Edge images of the two LDRIs shown in (a) and (b). (e)(f) Bitmap images of two LDRIs after applying the MTB transformation.*

The alignment of a sequence of LDRIs is implemented as follows. The middle exposure is chosen as the ground truth; all other exposures are aligned with respect to this exposure. The middle exposure I_m or at least the exposure captured in the middle of the exposure sequence is in general the best aligned with all other exposure in the sequence. Each exposure I_i ($i \neq m$) is aligned with the middle exposure I_m, using a binary image tree, similar to described in [War04]. The binary image tree of size L (L=4 in our case) is constructed as follows. The original images $I_i = I_i^0$ and $I_m = I_m^0$ reside at the lowest level (l=0). At the other levels $l \in [1, L]$, the images I_i^l and I_m^l are down-sampled versions of the original images with a down-sample factor equal to 2^l. The images I_i^l ($i \neq m$) and I_m^l are first aligned at level $l = L$. The calculated transformation is used as a start seed at level $l = L - 1$, where a new transformation matrix is calculated based on the images with down-sample factor 2^{L-1}. This process is repeated until $l = 0$. At a certain level l the best transformation $T(\cdot)$ (rotation and translation combined) returns the minimum difference between the binary images resulting from applying the MTB procedure on I_m^l and on the transformed image $T(I_i^l)$. The optimal transformation $T(\cdot)$ is found as the minimum of a set possible transformations. First the optimal translation $[T_x, T_y]$ (in steps of one pixel) is found, followed by the best rotation α (in steps of 0.5 degrees), and this process is iterated until the error converges. The search for this minimum can fail due to local minimum, but is less likely to get stuck in a local minima than when no binary tree is used.

The stability of the MTB alignment method suffers from noisy pixel intensities around MIV, which have an undefined influence on the binary threshold image [War04]. This instability can effectively be controlled by withholding the noisy pixel intensities from the alignment procedure, i.e., by excluding pixel intensities that lie within a certain range of MIV.

Just like with any other type of alignment algorithm, a moving object does not disturb the alignment if it does not create prominent features. A prominent feature is for instance an edge, when considering edge alignment, or a corner, when considering corner alignment. Such a prominent feature will interfere with the error function that is minimized to find the most suitable transformation. In our case, the moving object should not disturb the features after the MTB transform. One of the requirements could therefore be that the projection of the moving object onto the image should be small compared to the image dimensions. However, if the moving object has irradiance values smaller (or larger) than MIV, and its background is also smaller (or larger) than MIV, then the size of the projected object is less important, as it does not create any new features after the MTB transform.

The method presented here improves the limited alignment from Photosphere [Any], which uses the alignment presented in [War04], by also recovering rotational misalignments. Although in some situations a perspective alignment would improve the alignment, in our experience misalignments due to translation and rotation are usually the most dominant sources of misalignments.

5.6 Object movement removal

It is very difficult to maintain a static scene in a real-world scenario. Humans, cars, and windy conditions are sources of movement when capturing HDRIs. For relighting applications, this movement is a hazard if it occurs in front of important scene points, such as a light source. For this thesis we analysed the existing HDRI generation software, and to the best of our knowledge, none of the existing methods provides any object movement removal. This is at the exception of Photosphere [Any] that provides limited object movement removal, used in this chapter for benchmarking purposes.

Throughout our experiments, we realized that certain types of movement remain undetectable with the current implementation of movement detection of Photosphere in the context of HDR imaging. In fact,

we built a classification of types of movement and identified that while the existing method is successful for the first group of movement, it fails on the second type of movement. The first group comprises high contrast movement, the second group consists of low contrast movement. After having analysed the behaviour of the second type of movement, we developed a new movement detector. The main advantage of this detector is that it does not require the knowledge of the camera curve, while the first detector does. This has the advantage that the camera curve calibration could potentially use this information, and discard those pixels affected by movement from the camera curve calibration process.

Though this new movement detector can be applied to both categories of movement, we still find it useful to distinguish between the two categories prior to HDR merging, as the first method is considerably faster than the second. The strategy of the movement removal remains the same for both types of movement, and it is only the movement detection that is different.

This section is organized as follows. Section 5.6.1 provides a general framework for movement removal, which consists of a movement detection and movement substitution phase, and is independent from the specifications of the movement detection. Section 5.6.2 discusses our classification of movement into two groups. Section 5.6.3 explains the movement removal algorithm based on the implementation used in Photosphere [Any] and discusses its limitations. This movement removal process is also explained in [RWPD05], though our implementation might slightly be different than described there. Finally section 5.6.4 presents a new type of movement removal.

5.6.1 Movement Removal

The movement removal process follows the strategy outlined in figure 5.4, and is in principle independent from the type of movement detector. The movement removal process fits in the system presented in figure 5.2. For reasons of simplicity we have assumed, in figure 5.4 and in the discussion that follows, that only one moving object is present in the scene.

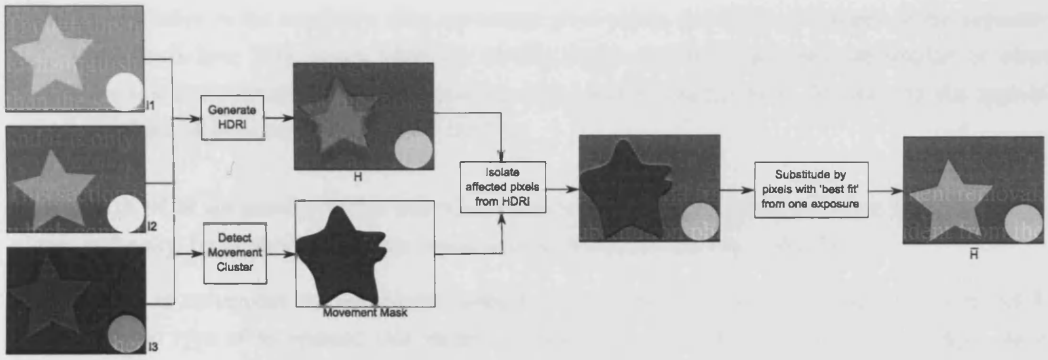


Figure 5.4: *The three LDRIs I_1, I_2 and I_3 contain a moving object (the star). As a result the star in the final HDRI H shows (usually undesired) ghosting effects. The movement detector identifies the cluster of pixels affected by this movement and creates a binary* movement mask, *where black indicates movement. This cluster is removed from H and substituted by the irradiance values from one exposure, I_2 in this example. The choice of exposure depends on the level of saturation that the moving object shows in each of the LDRIs. The final HDRI $\overline{H}$ is free from ghosting effects.*

After the HDRI generation, the moving object creates a ghosting effect in H. This is a direct result of the weighted merge of the LDRIs. As is discussed later, the movement detection operates on the LDRIs I_i and creates a binary *movement mask* which has the same dimensions as H. This movement mask defines *movement clusters* consisting of *movement pixels* (black), which are pixels affected by movement, and *non-movement pixels* (white). The generated HDRI H is multiplied with the movement mask to segment

the movement pixels from the other HDRI pixels. The irradiance covered by the movement cluster is analysed and one exposure I_i is chosen that represents the pixels inside the movement cluster with the least saturation. This exposure is then used to substitute the movement pixels in H to create a new HDRI $\overline{H}$. This HDRI $\overline{H}$ is now effectively free from ghosting effects.

If more than one moving object is present in the scene, and the moving objects are located at different positions in the viewing window, the movement mask should show different movement clusters for each of the moving objects. Each cluster should effectively cover the entire area of pixels affected by the moving object it is associated with. If the movement detector fails to do this, the substitution results in inconsistencies: some parts of the moving object are correctly extracted from one exposure, other parts still show some ghosting effects. If the sole objective is to create movement-free HDRIs, one might say that it is better to have clusters rather too larger than too small.

Substituting the pixels in H by the irradiance values retrieved from one LDRI is not error free. If the substituted pixels are saturated or under-exposed in the LDRI, the final HDRI $\overline{H}$ also contains saturation or under-exposure. Moreover, the merge of several LDRIs into one HDRI decreases the noise that might be present in one LDRI [DM97], therefore representing pixels with irradiance values from one exposure will most likely increase the image noise. Therefore choosing to substitute pixels in an HDRI by one exposure should be carried out with care.

5.6.2 Types of movement

When an object moves through a scene, two situations occur:

- The variation in the irradiance of a movement pixel across the different images of the exposure sequence is high. This occurs when there is a high contrast between the moving object and the background. We refer to this type of movement as *high contrast movement (HCM)*.
- The variation in the irradiance of a movement pixel across the different images of the exposure sequence is low. This occurs when the moving object and the background are similar, or when there is a low contrast between the moving object and the background. We refer to this type of movement as *low contrast movement (LCM)*.

Examples of HCM are moving people and cars against a fixed background, a flag flying in the wind, and planes in the sky. Examples of LCM are leaves and tree branches moving in the wind.

The first type of movement can be detected using a variance measure, as is explained in section 5.6.3. For the second type of movement, this variance measure is a less robust movement indicator. As a solution, we used entropy to provide a measure of uncertainty about a pixel's static status, as is discussed in section 5.6.4.

5.6.3 High contrast movement removal using a variance measure

A Weighted variance as a measure of movement

The pixels affected by the HCM, show a large irradiance variation over the different LDRIs. Therefore, the variance of a pixel over the different I_i's can be used as a likelihood measure for movement. The movement mask, defined in section 5.6.1, is derived from a *Variance Image (VI)*, which is created by storing the variance of a pixel over the different exposures in a matrix with the same dimensions as the images I_i and H.

Remember that the LDRIs are initially captured as low dynamic range images. Using the camera curve, the intensity values in the images I_i are transformed to irradiance values, resulting in images E_i. When using the variance of a pixel over a set of images to detect movement, this irradiance transformation is necessary, and failing to do so results in high variance values for all pixels, due to the implicit differences between the exposures. In mathematical terms we can now define the variance of a pixel (k, l) over the different exposures as:

$$VI(k,l) = \frac{\sum_{i=0}^{N-1} (E_i(k,l) - \overline{E(k,l)})^2}{N} \tag{5.1}$$

with:

$$\overline{E(k,l)} = \frac{\sum_{i=0}^{N-1} E_i(k,l)}{N} \tag{5.2}$$

Some pixels in the I_i's will be saturated, others can be under-exposed. Such pixels do not contain any reliable irradiance information, compared to their counterparts in the other exposures. When calculating the variance of a pixel over a set of images, it is important to ignore the variance introduced by saturated or under-exposed pixels. This can be achieved by calculating the variance $VI(k, l)$ as a weighted variance described in [RWPD05] as:

$$VI(k,l) = \frac{\frac{\sum_{i=0}^{N} W_i(k,l) E_i(k,l)^2}{\sum_{i=0}^{N} W_i(k,l)}}{\frac{(\sum_{i=0}^{N} W_i(k,l) E_i(k,l))^2}{(\sum_{i=0}^{N} W_i(k,l))^2}} - 1 \tag{5.3}$$

The weights are the same as defined in equation 2.32. The generation of VI is visualized in figure 5.5. The variance image can be calculated for one colour channel or as the maximum of the variance over three colour channels. In this chapter we calculated the variance over the green channel.

B Movement segmentation

The objective of creating VI is to identify clusters of pixels that show movement over the time period that the LDRIs were captured. In H such movement cluster are substituted with values from one exposure. As was discussed in section 5.6.1, each movement cluster should cover all pixels in H affected by the movement of one (or more) object(s). This can be achieved by simply applying a threshold T_{VI} to the values in VI, resulting in a binary image VI_T. However, due to noise, saturation effects, and an inappropriate camera curve, applying a threshold does not return well-defined clusters of movement. Some pixels might fall incorrectly below (false positives) or above (false negatives) the threshold. To create well-defined clusters, two morphological post-processing steps proceed after applying the threshold: first an erosion (to remove false positives), then a dilation (to include the falls negatives).

We found the variance threshold $T_{VI} = 0.18$ [RWPD05] to be efficient for the motion detection. The dimensions for the erosion and dilation filters depend on the resolution of the input image, but also on the quality of the image alignment as this might introduce unwanted high variance values. We found a

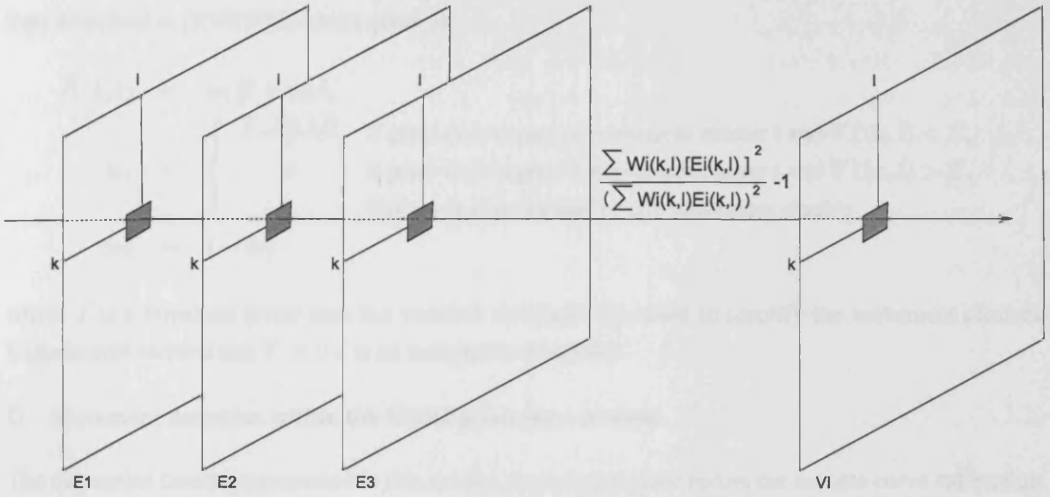


Figure 5.5: *The Variance Image (VI) is created by calculating the weighted variance of a pixel's irradiance over the different exposures.*

filter size of 5 and 10 for respectively the erosion and the dilation usually to be sufficient for images with a resolution of 1600×1200.

C Interpolated substitution

To find the best exposure to substitute the pixels in a cluster, the same method is applied as described in [RWPD05] which is explained in this subsection. For each cluster a histogram is generated from its values in H. The largest value $\widehat{x}$ represented in this histogram is calculated, after ignoring the 2% largest values since they can be considered as being outliers [RWPD05]. The exposure that correctly represents this value $\widehat{x}$ is used to substitute the pixels in H. Correctly in this context means that the value must not be saturated or under-exposed. For instance, the intensity value resulting from transforming the pixel with irradiance value $\widehat{x}$ conform with the exposure settings used for the exposures, must represent this intensity value in the middle range of its histogram.

In [RWPD05] it is stated that if more than one suitable exposures is found, the one with the longest exposure time is used to represent the movement cluster. We argue that it is better to take the shortest exposure time. If the exposures are affected by movement, the exposure with the smallest exposure time is less likely to be affected by the movement *during* the exposure of the camera sensor, and will most likely not show blurring. Hence, it is the most likely to represent its intensity values correctly.

When substituting the movement cluster, boundary effects can be created. Remember that the clusters are created from the binary image VI_T proceeded by dilation and erosion. As a consequence, movement pixels are grouped into a well-defined cluster. However, due to these morphological operations, the clusters will also contain non-movement pixels (false positives). To ensure that the irradiance of these pixels with an extreme low variance are not substituted and to reduce boundary effects, the pixels within a cluster are substituted using interpolation. A pixel in $\overline{H}$ within a movement cluster is defined as a weighted average between its irradiance value in the associated exposure E_i and its original value in H. The weights are defined by its variance. The weighting scheme used in this chapter is slightly different

than described in [RWPD05] and is given as:

$$\begin{aligned}
\overline{H}(k,l) &= w_1 H + w_2 I_i \\
w_1 &= \begin{cases} \frac{T - VI(k,l)}{T} & \text{if pixel (k,l) is part of movement cluster i and } VI(k,l) < T, \\ 0 & \text{if pixel (k,l) is part of movement cluster i and } VI(k,l) > T, \\ 1 & \text{if pixel (k,l) is not part of any movement cluster,} \end{cases} \\
w_2 &= 1 - w_1
\end{aligned}$$

where T is a threshold lower than the variance threshold T_{VI} used to identify the movement clusters. Experiments showed that $T = 0.1$ is an acceptable threshold.

D Movement detection within the HDRI generation process

The movement detection presented in this section cannot take place before the camera curve calibration, but takes place after the (incorrect) HDRI generation. The methodology presented in figure 5.2 still applies, and can be updated with the movement detection step as displayed in figure 5.6.

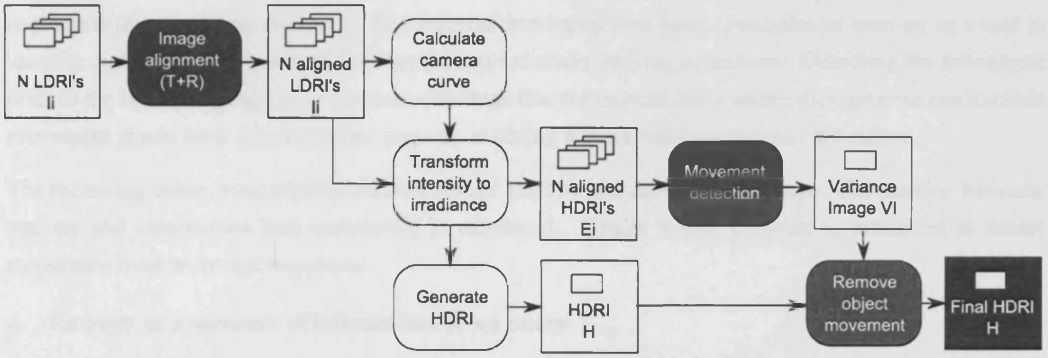


Figure 5.6: *HDRI generation methodology for dynamic scenes. The variance image calculation must proceed after the camera curve calibration and the transformation from intensity to irradiance.*

The movement detection and creation of the variance image precede the movement removal but follows after the transformation from intensities to irradiance values. The fact that the movement detection occurs after the camera curve calibration means that it is essential that the camera curve calibration is not compromised by the object movement. This is a serious drawback from this method. The following section presents a novel movement detector, developed for this thesis, that operates without using the camera curve.

5.6.4 Contrast independent movement removal using entropy

When applying the previous variance-based movement detector on some example exposure sequences, we identified that it is not always successful. The variance detector returns unsatisfying results, in particular when the movement has a low contrast nature. The variance detector relies on a threshold T_{VI} that defines if a pixel has considerable variance or not. This threshold is larger than zero, and depends on:

- Camera curve: the camera curve might fail to convert the intensity values to irradiance values correctly. This influences the variance between corresponding pixels in the LDRIs and might compromise the applicability of the threshold to retrieve movement clusters.

- Weighting factors: saturation and under-exposure of pixels in an LDRI can result in incorrect irradiance values after transformation to irradiance values using the camera curve. Defining the weighting factors is not straightforward. Applying a threshold to remove saturated and under-exposed pixels, for instance when using a hat function to define the weights, is not necessarily successful. Though [MN99] applies a more sophisticated scheme, based on information theory, the weights defined do not guarantee a 100% blocking of saturation/under-exposure effects.
- Inaccuracies in exposure speed and aperture width: in combination with the camera curve this produces incorrect irradiance values after transformation. Changing the aperture width causes the depth-to-field to change too, which influences the quality of the irradiance values.

Relying on the fact that the camera curve transforms correctly the intensity images I_i to irradiance images E_i can be seen as a limitation of the variance detector. Though it is true that if the camera curve does not transform correctly the intensities to irradiance values the HDRIs do not represent correctly the environment, there might be applications for which small errors in HDR values might not be disastrous.

In this section a method is presented that does not require the transformation from intensity to irradiance to precede the movement detection. The method developed uses basic principles as entropy as a tool to identify movement in a sequence of images captured under varying conditions. Detecting the movement prior to the HDR merging has as an extra advantage that the camera curve calibration process can exclude movement pixels from the estimation process, enabling a more reliable camera calibration.

The following subsections explain the concepts of entropy and conditional entropy. The relation between entropy and information and uncertainty is discussed. Finally a new measure is presented to detect movement from an image sequence.

A Entropy as a measure of information in an image

Entropy is a term used in both thermodynamics and information theory. In this chapter we consider the definitions and concepts of entropy in the field of information theory, as defined by Claude E. Shannon [Sha84].

In information theory, entropy is a scalar statistical measure defined for a statistical process. It defines the uncertainty that remains about a system, after having taken into account the observable properties. Let X be a random variable with probability function $p(x) = P(X = x)$, where x ranges over a certain interval. The entropy of X is given by:

$$H(X) = -\sum_{x} P(X = x) \log(P(X = x)). \tag{5.4}$$

Entropy is often denoted with capital H, this should not be confused with the abbreviation of HDRI H. From the context in which it is used, it should be clear which of the two concepts is meant.

To derive the entropy of an image I, written as $H(I)$, we consider the intensity of a pixel in an image as a statistical process. In other words, X is the intensity value of a pixel, and $p(x) = P(X = x)$ is the probability that a pixel has intensity x. The probability function $p(x) = P(X = x)$ is the normalized histogram of the image. Normalized means that the sum of the probabilities needs to be one. Therefore we divide the histogram values by the total number of pixels in the image. The pixel intensities range over a discrete interval, usually defined as the integers in $[0, 255]$, but the *number of bins* M of the histogram used to calculate the entropy can be less than 256[1].

[1] It could be larger too, but that would not make sense, as an image usually contains integer values from 0 to 255.

If the histogram of an image I is flat, then all intensities are present in the image in equal amounts and the entropy $H(I)$ is:

$$\begin{aligned} H(I) &= -\sum_{x=0}^{M} P(X=x)\log(P(X=x)) \\ &= -\sum_{x=0}^{M} \frac{1}{M}\log\frac{1}{M} \\ &= -\sum_{x=0}^{M} \frac{1}{M}(\log(1)-\log(M)) \\ &= \sum_{x=0}^{M} \frac{\log(M)}{M} \\ &= M\frac{\log(M)}{M} = \log(M) \end{aligned} \tag{5.5}$$

This is the maximum value that the entropy of an image can possibly have, for a certain number of bins M. The higher the number of bins M used to calculate the probabilities, the higher the maximum entropy. This is logical since the number of different values X can have is proportional to the number of bins M. Therefore the uncertainty about the pixels in an image is dependent on the number of bins M used.

The entropy reaches a minimum if an image is homogenous and consists of only one intensity value Z. In that case the histogram has a spike near that intensity value and is zero elsewhere: the probability of a pixel having intensity value Z is one, and zero for all other possible intensity values. The entropy $H(I)$ is now:

$$\begin{aligned} H(I) &= -\sum_{x=0}^{M} P(X=x)\log(P(X=x)) \\ &= -P(X=Z)\log(P(X=Z)) \\ &= -1\log 1 \\ &= 0 \end{aligned} \tag{5.6}$$

Note that the above equations are independent from the actual value of Z. Let us consider another example. This time the pixels in the image can have two different values Z_1 and Z_2. Suppose the likelihood of a pixel having any of these values is equal, or that $P(X=Z_1) = P(X=Z_2) = \frac{1}{2}$. The entropy of this image can be written as:

$$\begin{aligned} H(I) &= -\sum_{x=0}^{M} P(X=x)\log(P(X=x)) \\ &= -P(X=Z_1)\log(P(X=Z_1)) - P(X=Z_2)\log(P(X=Z_2)) \\ &= -\frac{1}{2}\log(\frac{1}{2}) - \frac{1}{2}\log(\frac{1}{2}) \\ &= log(2) \end{aligned} \tag{5.7}$$

Again, the entropy of image I is independent of the actual values of Z_1 and Z_2. The entropy of an image I is the same for $Z_1 \gg Z_2$ and $Z_1 \simeq Z_2$, as long as $Z_1 \neq Z_2$. In more general terms this means that the contrast in the image, defined as $\frac{Z_1}{Z_2}$, does not influence the entropy.

Equations 5.5, 5.6 and 5.7 show several important characteristics of entropy of an intensity image:

- The entropy of an image has a positive value between $[0, \log(M)]$. The lower the entropy, the less different intensity values are present in the image; the higher the entropy, the more different intensity values there are in the image. However, the actual intensity values do not have an influence on the entropy.
- The actual *order* or *organization* of the pixel intensities in an image does not influence the entropy. As an example, two images with equal amounts of black and white intensity values have the same entropy, even if in the first image black occupies the right side of the image and white the left side, and in the second image black and white are randomly distributed.
- Applying a scaling factor on the intensity values of an image does not change its entropy, if the intensity values do not saturate. In fact, the entropy of an image does not change if an *injective function* is applied to the intensity values. An injective function associates distinct arguments to distinct values, examples are the logarithm, exponential, scaling, etc.
- The entropy of an image gives a measure of the uncertainty of the pixels in the image. If all intensity values are equal, the entropy is zero and there is no uncertainty about the intensity value a random pixel can have. If all intensity values are different, the entropy is high and there is a lot of uncertainty about the intensity value of a randomly chosen pixel.
- Entropy of a data stream is often used to encode data. In a similar manner the entropy in an image gives a measure for the amount of information in an image. If an image contains many different intensity values, it contains a lot of information ($H(I)$ is high); if an image is homogeneous, it contains very little information ($H(I)$ is low).

The concept *joint entropy* $H(X,Y)$ of two stochastic processes X and Y gives information about the mutual information between X and Y. It is again a scalar, and its definition is given as:

$$H(X,Y) = -\sum_{x,y} P(X=x, Y=y) \log(P(X=x, Y=y)) \tag{5.8}$$

The joint entropy of two images I and J (with the same dimensions $N \times M$) can be calculated from their joint probability $P(X=x, Y=y)$ which is equal to their joint histogram. $P(X=x, Y=y)$ defines the occurrence that a pixel in image I has intensity x and that its corresponding pixel in image J has intensity y.

The joint probability is a 2D function. When image I and J are equal, the joint probability can be reduced to a 1D function equal to the histogram of one of the images, since:

$$P(X=x, Y=y) = \begin{cases} 0 & \text{if x}\neq\text{y} \\ P(X=x) & \text{if x=y} \end{cases}$$

Suppose that the only difference between image I and image J is a scale S on their pixel values, then the joint probability can again be reduced to the same 1D function (though rotated in the 2D coordinate space):

$$P(X=x, Y=y) = \begin{cases} 0 & \text{if x} \neq \text{Sy} \\ P(X=x) & \text{if x=Sy} \end{cases}$$

If two processes X and Y are independent, the joint entropy $H(X,Y)$ is equal to $H(X)$ + $H(Y)$. If X and Y are equal, the joint entropy is equal to $H(X)$. From the definition of joint entropy and the above given definitions and different cases, the following remarks can be made for joint entropy:

- For an image I, the joint entropy with another image J has a value in the interval $[min(H(I), H(J)), H(I) + H(J)]$. The lower the joint entropy the more similar the two images are; the higher the joint entropy, the more different the two images are.
- A scale on the intensity values in one image does not influence the joint entropy. In fact, applying an injective function on the intensity values of one image, does not change the joint entropy.

The *conditional entropy* $H(X|Y)$ is given as:

$$\begin{aligned} H(X|Y) &= -\sum_{x,y} P(X=x|Y=y)\log(P(X=x|Y=y)) && (5.9)\\ &= H(X,Y) - H(Y) && (5.10)\end{aligned}$$

The conditional entropy $H(I|J)$ of two images I and J gives a measure of the uncertainty there is about the intensity value of a randomly chosen pixel in image I, given the intensity value of the corresponding pixel in J. If two images are related, this uncertainty is rather low. If two images are totally unrelated, this uncertainty is high. For the latter case, one can say that the knowledge of image J does not learn anything about image I.

The *mutual information* $MI(X,Y)$ between two processes X and Y is given as:

$$\begin{aligned} MI(X,Y) &= H(X) + H(Y) - H(X,Y)\\ &= H(X) - H(X|Y) \leq H(X)\\ &= H(Y) - H(Y|X) \leq H(Y)\\ &\geq 0 && (5.11)\end{aligned}$$

The mutual information can also be defined for images I and J. If the two images are related, i.e., $H(I) \approx H(J)$ and $H(I,J) \approx H(I)$, the mutual information is high:

$$MI(I,J) \approx H(I) \tag{5.12}$$

If the two images are unrelated, i.e., $H(I) \neq H(J)$ and $H(I,J) \approx H(I)+H(J)$, the mutual information is low:

$$MI(I,J) \approx 0 \tag{5.13}$$

B Detecting misalignments in an image using entropy

From the discussions given in the previous subsection it follows that maximization of mutual information between two images is a good statistic to find the best transformation between two images [VW95][Vio95][TU00]. The advantage of using mutual information for image alignment compared to any other traditional correlation-based alignment methods is that there does not need to be a linear relation between the two images. Non-linear transformations such as a logarithmic transformation are also allowed. As long as the viewpoint and the scene content stay the same, alignment is feasible. Therefore, mutual information is an ideal statistical measure to align, for instance, an ordinary image and an infra-red image. Though seemingly different, the scene content is essentially the same, and therefore the two images can be aligned using mutual information. In the latter example, a feature-based method might be successful too, but template-based matching is not. Costers and Jacobs [CJ02] illustrate the use of mutual information to align sequences of images under varying illumination.

The different exposures captured for the HDRI generation are related by the camera curve. Equation 2.31 already defined this relation. The shape of a camera curve does not change much between different cameras and according to [GN03b] the general curve of a camera is monotonic and increasing. The curve is usually injective in the middle of the domain, while this statement is not true near the ends of the curve due to the saturation and under-exposure effects. Mutual information methods or methods using entropy are therefore suitable to align exposure sequences, if saturation effects can be ignored or removed.

In this chapter we use the MTB transform to aid the alignment process, instead of mutual information. The reason is that it is computationally more attractive as it requires less calculations than alignment through maximisation of mutual information, which requires expensive histogram calculations.

C Movement detection

The movement detection algorithm developed in this chapter has some resemblance to that presented in [MZ01] and [JSR04]. In [MZ01] a Spatial-Temporal Entropy Image (STEI) is used to detect motion in a sequence of images. In [JSR04] the creation of the STEI is updated to a Differential Spatial-Temporal Entropy Image (DSTEI). These methods work for images captured under the same conditions, i.e., using the same exposure and from the same viewpoint, and with static illumination.

Because these two methods lie at the basis of the method developed in this chapter, it is worth giving more information about the generation of them, in particular the STEI image. The STEI image has the same size as the N images I_i in the sequence from which motion needs to be detected. It is created by calculating the *local entropy* for each pixel $STEI(k,l)$ with image coordinates (k,l). Entropy is defined over a statistical variable. In their approach this variable is still the intensity of a pixel, but the data set is now confined to the pixels in a small 3D window W around $STEI(k,l)$ and across all images of the image sequence. If the window size is $(2w+1) \times (2w+1) \times N$, the probability function is defined by the normalized histogram of the intensity values of the set of pixels p defined as:

$$\{\forall p \in I_i(k-w : k+w, l-w : l+w) \wedge i \in [0, N-1]\} \tag{5.14}$$

Suppose $N = 3$, if for a certain pixel p the three images show no movement in the window W around p, then the intensity values of the three images contribute the same information to the probability function. If the images show movement in the window W around p, the intensity values of the three images contribute in different ways to the probability function. In the latter case, it is more likely that the probability function is *flatter* (spread-out over the existing intensity values), than in the former case. As a result the entropy of p calculated as defined above will be higher when movement is present in the window W around p than when no movement is present.

Such a movement detection has some drawbacks, first of all other high *entropic* regions in the images (e.g., edges) might interfere with the motion regions, which makes it difficult to distinguish the high entropic clusters due to motion from those due to, for instance, edges. This problem is partly covered by the method described in [JSR04] through the creation of DSTEI which calculates the local entropy over the difference between the images in the image sequence. Another problem relates to the difference that might occur between the images in the image sequence besides movement. For instance, if the image sequence is captured under different illumination conditions, using a STEI or DSTEI image to detect movement is error-prone as it is essential for the success of these methods that the images are similar up to the movement that might take place. STEI and DSTEI can therefore not be used to calculate motion in an image sequence captured with different exposure settings.

The logical next step would be to use joint entropy between the images in the sequence, instead of

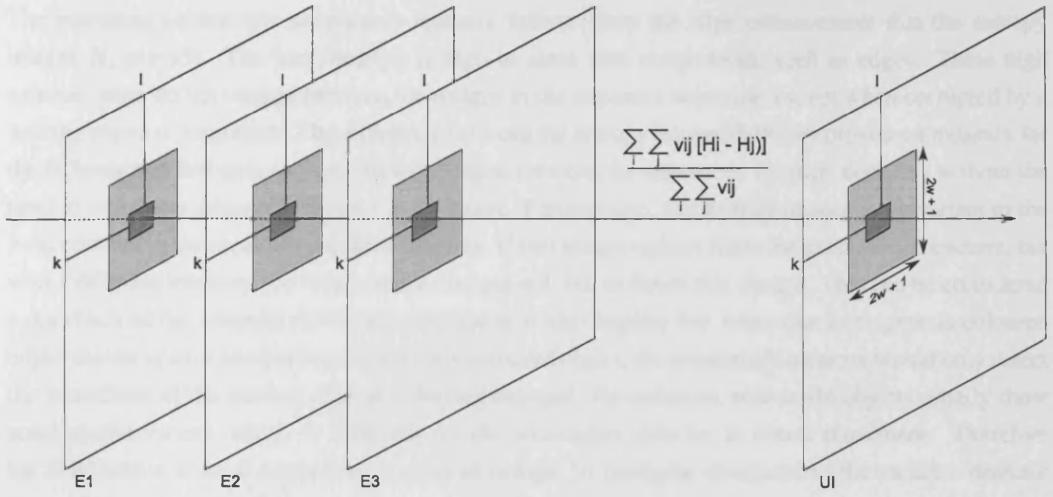


Figure 5.7: *The Uncertainty Image (UI) is created by calculating the per-pixel weighted difference in local entropy.*

plain entropy. The joint entropy over a certain 2D window is high when the images contain unexpected intensity values (motion) within that window. The advantage of using joint entropy is that it does not matter which exposure is used to capture the different images. However, using joint entropy was not satisfactory, most likely due to the small sample set (all pixels within a 2D window W) that is used to set up the statistic $P(x, y)$.

Another, this time successful, measure uses the difference in local entropy between the images. For each pixel with coordinates (k, l) in each image I_i the local entropy is calculated from the histograms constructed from the pixels that fall within a 2D window W with size $(2w+1) \times (2w+1)$ around (k, l). Each image I_i therefore defines an *entropy image* H_i, where the pixel value $H_i(k, l)$ is calculated as:

$$H_i(k,l) = -\sum_{x=0}^{M} P(X = x) \log(P(X = x))$$

where the probability function $P(X = x)$ is derived from the normalized histogram constructed from the intensity values of the pixels within the 2D window W, or over all pixels p in:

$$\{p \in I_i(k - w : k + w, l - w : l + w)\} \tag{5.15}$$

From these entropy images a final *Uncertainty Image* UI is constructed as follows:

$$UI(k,l) = \sum_{i=0}^{N-1} \sum_{j=0}^{j<i} \frac{v_{ij}}{\sum_{i=0}^{N-1} \sum_{j=0}^{j<i} v_{ij}} h_{ij}(k,l) \tag{5.16}$$

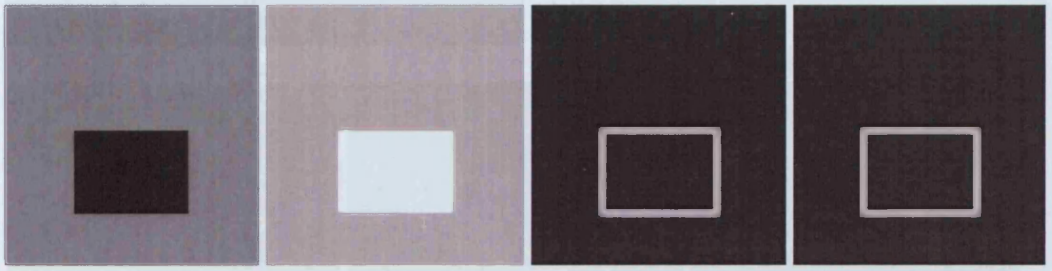
$$h_{ij}(k,l) = |H_i(k,l) - H_j(k,l)| \tag{5.17}$$

$$v_{ij} = min(W_i(k,l), W_j(k,l)) \tag{5.18}$$

The weights $W_i(k, l)$ and $W_j(k, l)$ are the same as those used in equation 2.32. The weight $v_{ij}$is created as the minimum of $W_i(k, l)$ and $W_j(k, l)$, to reflect the idea that saturated and under-exposed pixels do not provide a good statistic. The presented calculation process is also depicted in figure 5.7.

The reasoning behind this *uncertainty measure* follows from the edge enhancement that the entropy images H_i provide. The local entropy is high in areas with many detail, such as edges. These high entropic areas do not change between the images in the exposure sequence, except when corrupted by a moving object or saturation. The difference between the entropy images therefore provides a measure for the difference in features, such as intensity edges, between the exposures. Entropy does this without the need to search for edges and corners in an image. Furthermore, the entropy images are invariant to the local contrast in the areas around these features. If two image regions share the exact same structure, but with a different intensity, the local entropy images will fail to detect this change. This can be considered a drawback of the entropic movement detector as it also implies that when one homogenous coloured object moves against another homogeneously coloured object, the uncertainty measure would only detect the boundaries of the moving objects of having changed. Nevertheless, real-world objects usually show some spatial variety, which is sufficient for the uncertainty detector to detect movement. Therefore the indifference to local contrast is only an advantage, in particular compared to the variance detector discussed in section 5.6.3.

The characteristics and power of the uncertainty image are explained using two synthetic examples. A first synthetic example is presented in figure 5.8, which shows two images (a) and (b) with the same base structure: a square against a clear background. The contrast in these two images is different. The entropy image of (a) and (b) are shown in respectively (c) and (d). The entropy image is bright in areas with a high local entropy, which in this example is the border of the square against the background. The two entropy images (c) and (d) are identical which shows that the contrast of the scene does not influence the local entropy, but only the structure itself.



(a) (b) (c) (d)

Figure 5.8: *Entropy as a tool to detect structures. (a) and (b) show the same scene (a square object against a homogenous background), but with a different intensity pattern. The contrast between object and background is different in both cases. (c) and (d) are entropy images, generated by applying the local entropy operator on (a) and (b). The entropy operator enhances areas with a high local entropy (bright areas). (c) and (d) are identical, this illustrates that the local entropy operator is independent from the local contrast of the scene.*

Another example is shown in figure 5.9 which illustrates that equation 5.16 provides a good measure to detect object movement for a similar synthetic example. Images (a) and (b) show a similar scene that consists of a helicopter against a clear background. The position of the helicopter is different between (a) and (b), the intensity of the background is different too. Images (c) and (d) are the local entropy images of respectively (a) and (b). Finally, image (e) shows the uncertainty image derived from (a) and (b). This uncertainty image highlights area of movement. Note that the difference in intensity of the background did not result in a difference in local entropy in those areas. This illustrates the usefulness of the entropy detector in the context of different exposures. The helicopter shows an intensity pattern, this is to mimic the spatial variety that usually exists for objects in a real-world scene.

When no motion effects occur in a local window W around a pixel r, the local entropy in image I_i should be equal to the local entropy of image I_{i+1}. Noise and quantization effects might alter the entropy

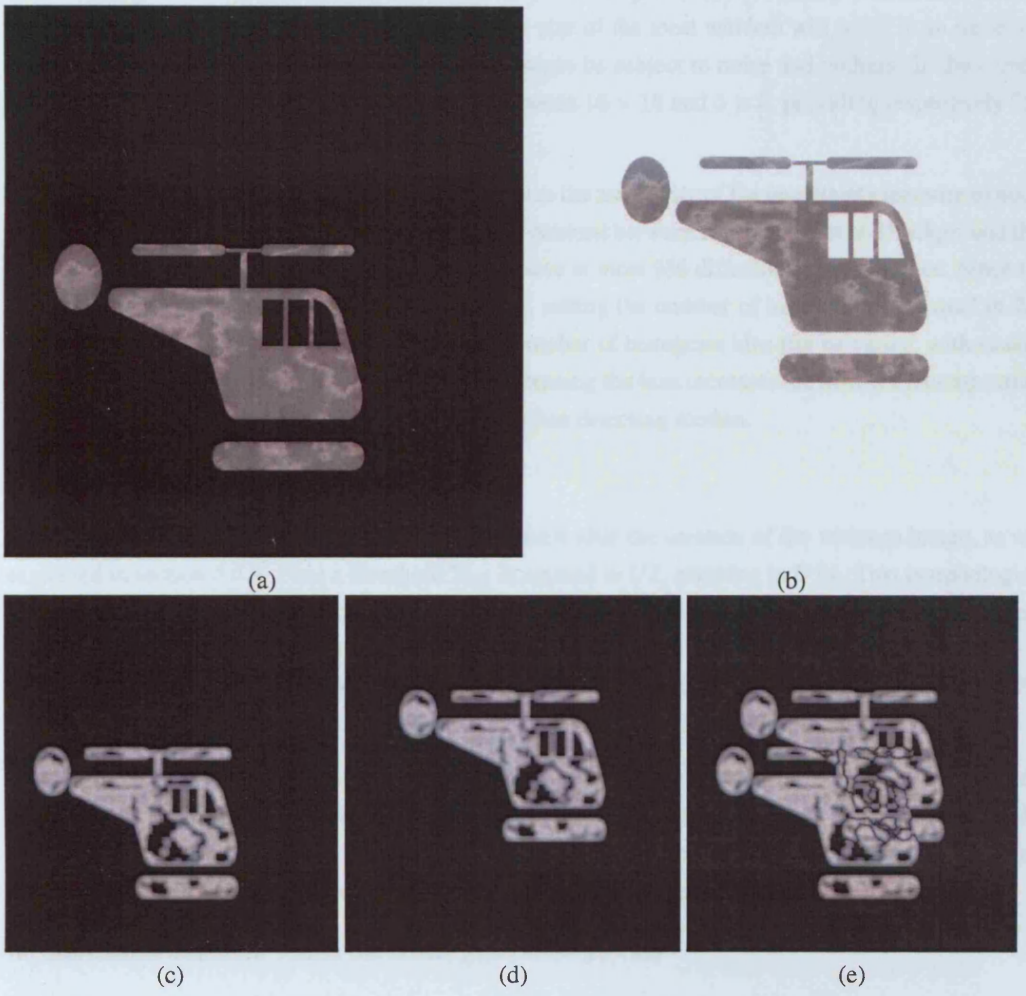


Figure 5.9: *(a) and (b) show a similar scene consisting of a helicopter against a clear background. The position of the helicopter is different between (a) and (b), the intensity of the background is different too. (c) and (d) are the local entropy images of respectively (a) and (b). (e) is the absolute difference $|c - d|$ and shows high uncertainty in the movement areas. These areas correspond to the areas showing movement.*

slightly, but these can be reduced through applying a low-pass filter to all exposures. A gaussian filter was used in our implementation. The weights in equation 5.16 should effectively remove saturation and under-exposure effects.

The difference in local entropy between two images induced by the moving object, depends on the difference in entropy of the moving object and the background environment. Though the uncertainty measure is invariant to the contrast of these two, it is not invariant to the entropic similarity of the two. For instance, if the local window is relatively large, the moving object is small relative to this window, and the background consists of many similar static smaller objects, then the entropic difference defined in equation 5.16 might not be large. Decreasing the size of the local window will result in an increased entropic difference, but a too small local window might be subject to noise and outliers. In the current implementation the optimal window size varies between 15×15 and 5×5, providing respectively 225 and 25 samples.

Decreasing the number of histogram bins might reduce the sensitivity of the uncertainty measure to noisy pixels, but will automatically increase the minimum contrast between moving object and background that can be detected. A pixel in an intensity image can have at most 256 different intensity values. Since the local window contains at most 225 different pixels, setting the number of histogram bins equal to 256 is sub-optimal. In the current implementation the number of histogram bins can be varied, with usually more than 100 bins providing satisfying results. Increasing the bins increases the histogram computation time, but increases the sensitivity to local contrast when detecting motion.

D Post-processing

The post-processing steps are similar to those involved after the creation of the variance image, as was explained in section 5.6.3. First a threshold T_{UI} is applied to UI, resulting in UI_T. Two morphological operations, erosion followed by dilation are applied to UI_T to obtain well-defined movement clusters. The radiance values of all pixels in each movement cluster in the thus obtained binary image UI_T is analysed and the most suitable exposure is used to substitute the pixels in the generated HDRI H. The most suitable exposure is defined in the same way as for the variance detector, see section 5.6.3.

The threshold applied to obtain UI_T depends on the number of histogram bins M used to calculate the local entropy. From equation 5.5 it can be derived that the maximum local entropy in an image is $\log M$. Choosing $UI_T = 0.7$ proofed to be a sufficient threshold to segment the movement clusters. The same filter sizes for the erosion and dilation were used as for the variance detector.

E Movement detection within the HDRI generation process

Due to the invariance of the detection process from the exposure settings used, the movement detection takes place before the camera curve calibration. The detection follows the image alignment immediately. The methodology presented in figure 5.2 still applies, and can be updated with the movement detection step as displayed in figure 5.10. This means that during the camera curve calibration, the uncertainty image UI can be used to exclude certain pixels from the camera curve estimation.

5.7 Results

The results are presented in the following sections. First, the performance of the MTB transform is analysed based on some examples in section 5.7.1. Then the variance detector and movement removal are applied to a few different movement scenarios in section 5.7.2. Section 5.7.3 shows the efficiency of the uncertainty detector and movement removal.

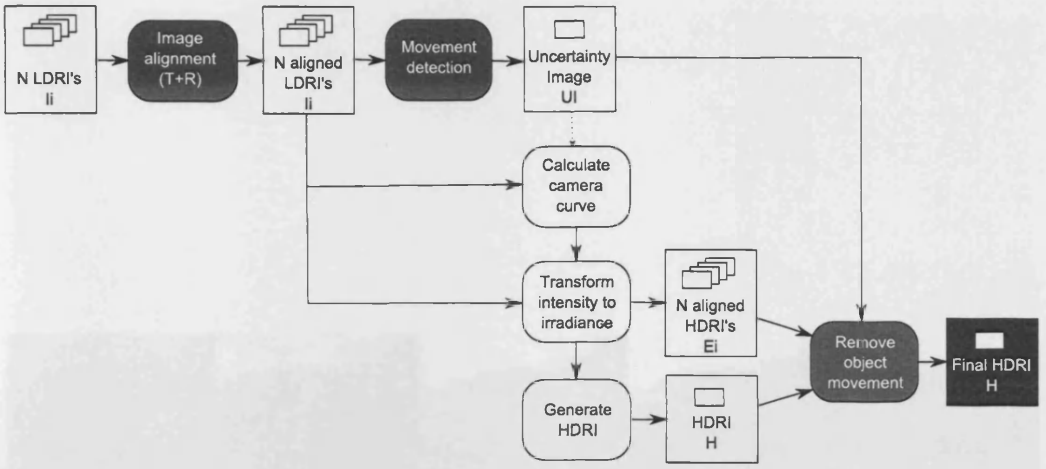


Figure 5.10: *HDRI generation methodology for dynamic scenes. The movement detection can be carried out independent from the camera calibration. If carried out before the camera curve calibration, the calibration could potentially benefit from knowing which pixels are affected by movement.*

5.7.1 Camera alignment

Figure 5.11 illustrates the binary image search tree, described in section 5.5, used to calculate the alignment transformation. The top row shows the exposures that require rotational and translational alignment. For two of these exposures the binary image tree is given, with size $L = 4$. First the images are aligned at level $l = 4$. The calculated transformation is exported to the level above, and used as a start seed to find the best transformation at level $l = 3$. This is repeated until $l = 0$. Before the down-sampling a low-pass filter is applied to the original images, to prevent aliasing. The binary images are created by applying the MTB transformation on the images at each level. Alignment using a binary tree as explained here, usually completed in less than one minute. With the alignment being dependent on the resolution of images, the binary tree search method gains speed at a certain level from the favourable starting point that is calculated at a higher level. This results in a more reliable overall alignment as non-hierarchical methods tend to get stuck in local minima more easily.

Figure 5.12 shows the HDRI generation when no alignment (a,d), translational alignment (b,e) and translational and rotational alignment (c,f) are carried out. The left column shows the entire image, the right column shows an image detail in close-up. The strange blue and pink colours visible in these close-ups are the result of the improper weighting of misaligned pixels during the HDRI generation. In (f), after recovering the translational and rotational transformation, the misalignments are the least visible.

Figure 5.13 (a) illustrates another HDRI generated from a set LDRIs captured with a hand-held camera. The different scene features are misaligned and create a ghosting effect in the HDRI. Images (b), (c) and (d) show close-ups of several details in (a). The ghosting effects are particularly visible on the mast of the ship as shown in (d).

Figure 5.14 shows the HDRI created from the same exposures as in figure 5.13, after removing translational (a) and translational plus rotational (b) misalignments. At first sight it is clear that the camera movement artefacts have been reduced in the two images, though the difference between the two might not be clear at this resolution. Close-ups for two image details in (a) and (b) are shown in figure 5.15. The images in the left column (a,c) belong to image (a) in figure 5.14; the images in the right column (b,d) belong to image (b) in figure 5.14.

Figure 5.16 illustrates an example of alignment failure. The four exposures shown in (a) are affected by

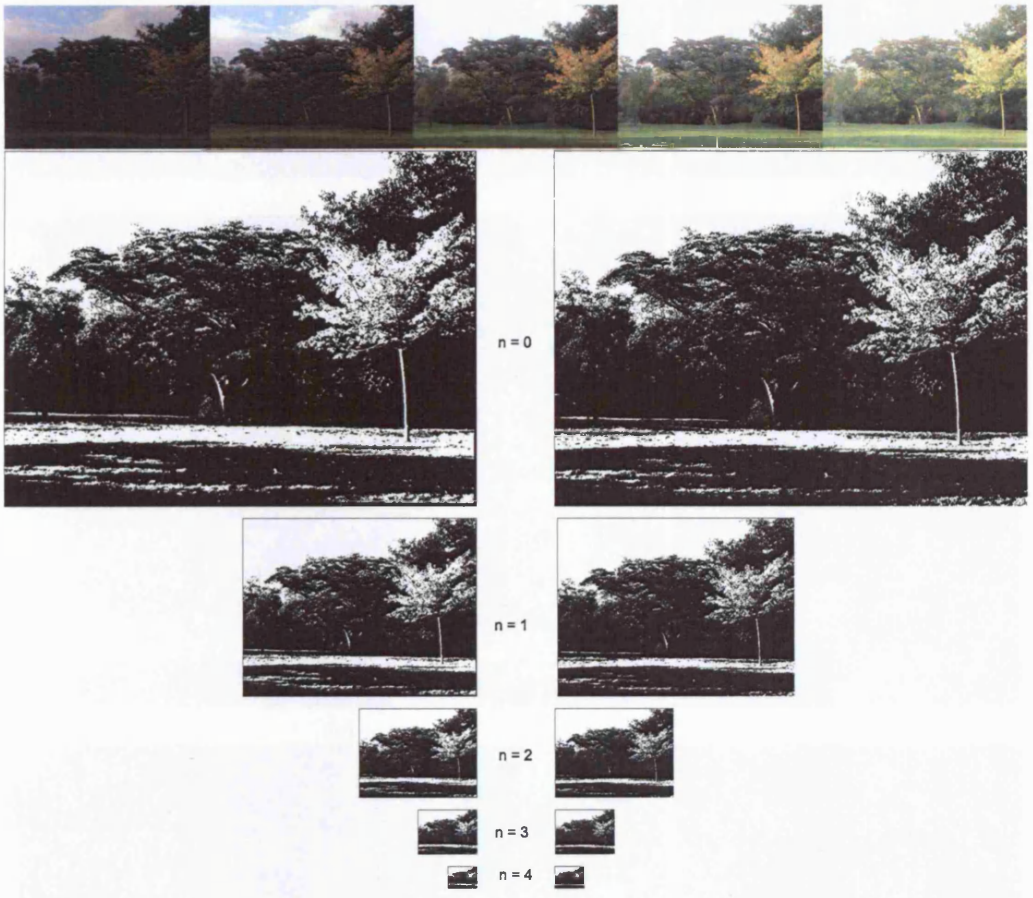


Figure 5.11: *Two binary image trees of two different exposures (left and right) from the set of exposures shown at the top. The tree depth is 5. The alignment procedure starts at the highest level n=5. The transformation from level n is used as a start seed at level n-1.*

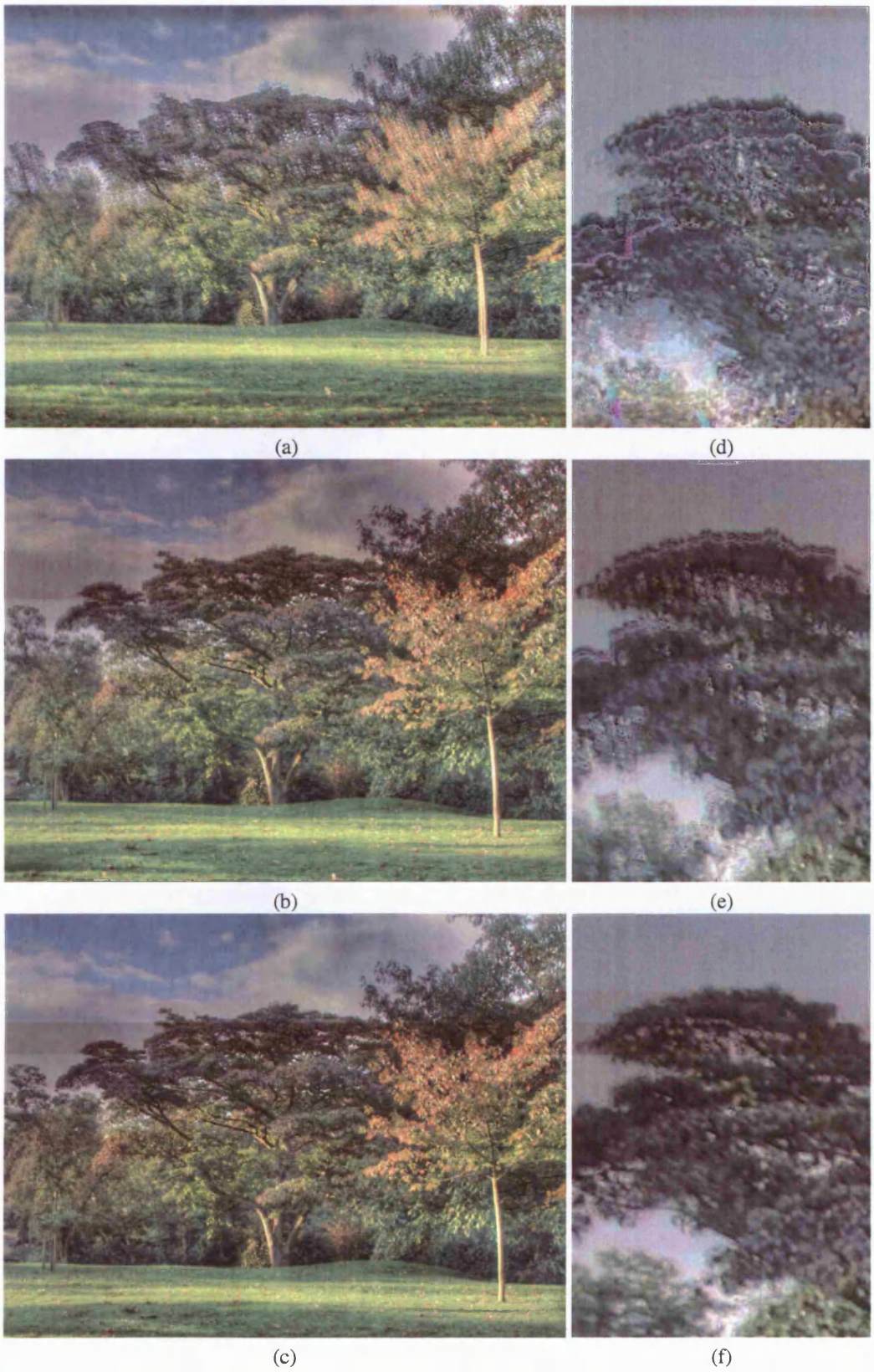


Figure 5.12: *HDRI generation and the influence of camera movement. The left column shows the entire HDRI, the right column shows an image detail in close-up for the following scenarios: no image alignment (a,d), translational alignment (b,e), translational and rotational alignment (c,f).*

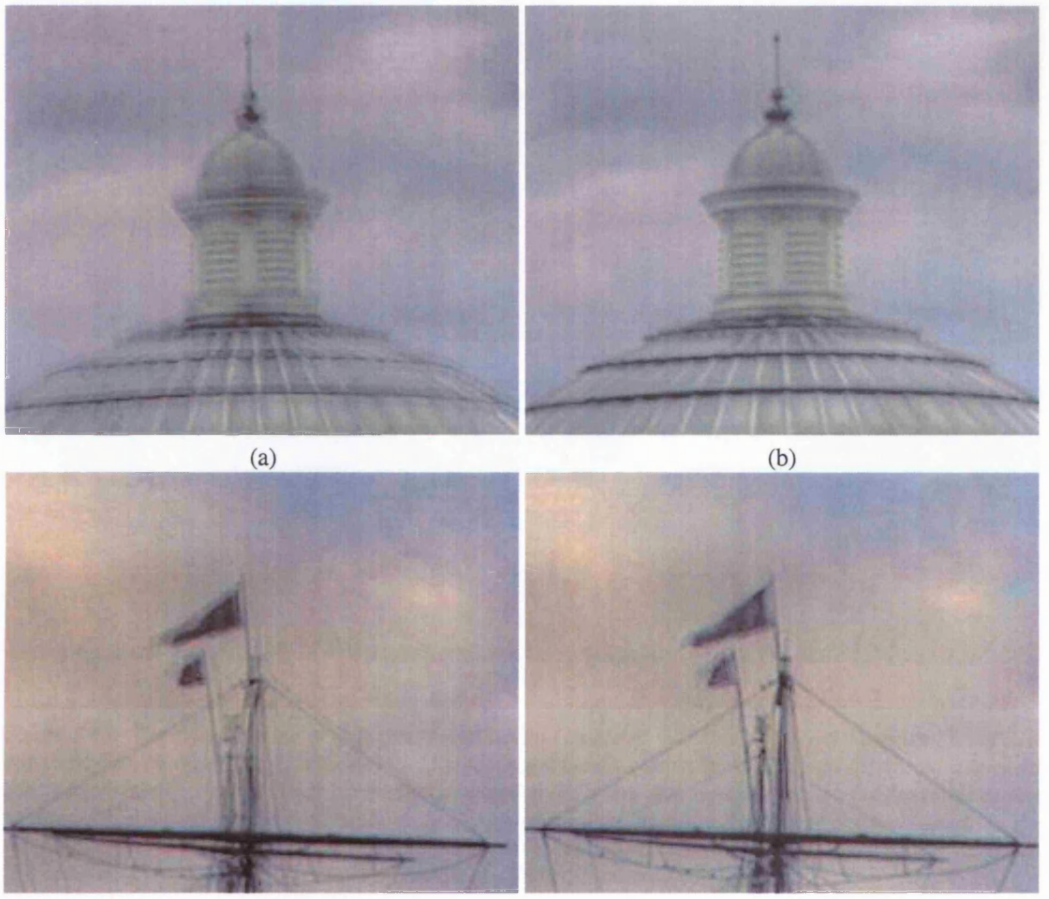
(a)

(b) (c) (d)

Figure 5.13: *(a) HDRI generated from a set of LDRIs captured with a hand-held camera. (b,c,d) Camera misalignments create ghosting effects, especially visible on the mast of the ship.*

(a) (b)

Figure 5.14: *The ghosting effects shown in figure 5.13 are reduced after recovering translational (a) and translational plus rotational (b) misalignments. See also figure 5.15*



(a) (b)

(c) (d)

Figure 5.15: *These images are close-ups from images (a) and (b) in 5.14. Recovering rotational and translational misalignments (b,d) improves the quality of the HDRI, compared to only recovering translation (a,c). The top of the dome in (a) still shows the misalignments due to the camera movement, while in (b) the alignment is much better. The ropes on the mast in (c) still show a ghosting effect, while it is removed in (d).*

camera movement but also by limited object movement. The tree branches on either side of the view window move in different directions relative to the camera viewing directions. The images should be aligned relative to the horizon, visible in the middle of the image. After alignment, the object movement could be removed using the algorithms explained in section 5.6. However, the object movement interferes with the camera alignment. The resulting HDRI is shown in (b) and a close-up of an image detail in (d). If we remove the second exposure from the sequence shown, alignment is obtained. The resulting HDRI is shown in (c) and a close-up of the same image detail in (e). The difference between (d) and (e) illustrates that the misalignment in the exposure sequence is the cause of the ghosting effect visible in the cloud, and not movement of the cloud during the exposure.

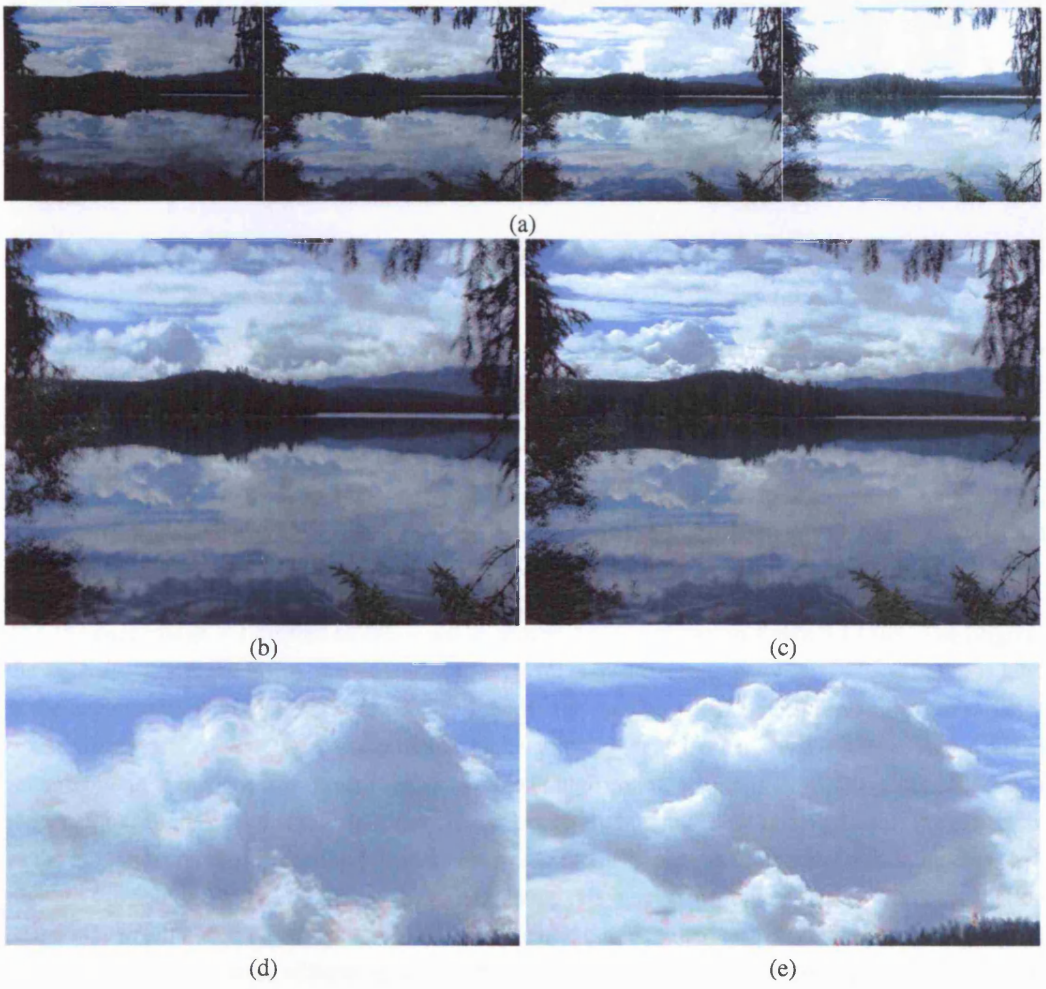
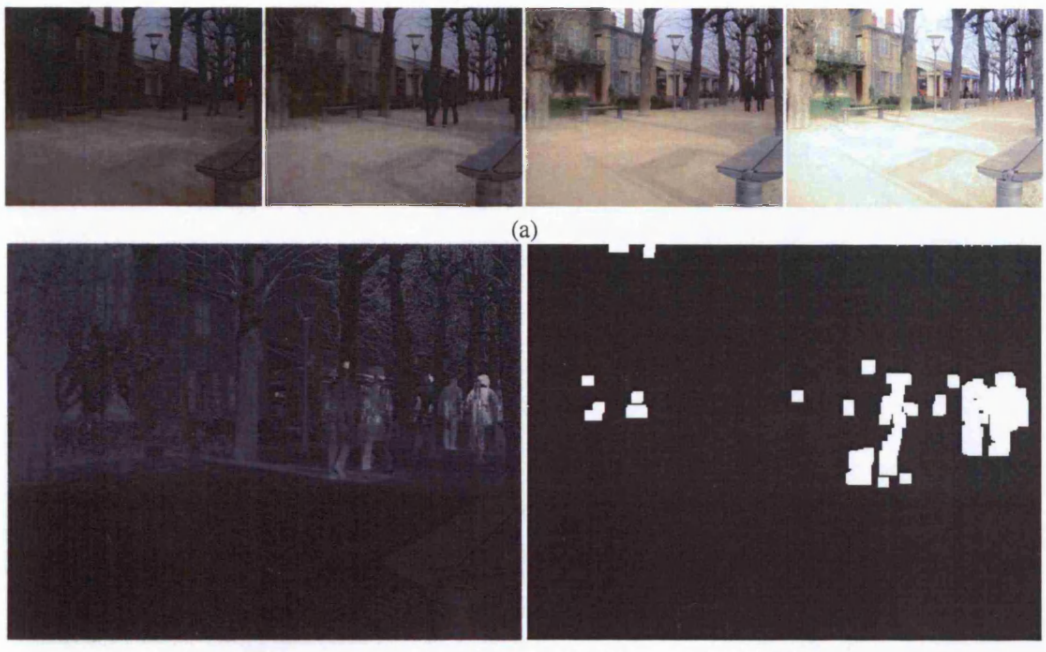


Figure 5.16: *The exposures shown in (a) are combined into an HDRI (b). Due to camera movement, some misalignments are visible. This might not be visible at first sight, but if we zoom in on the clouds (d), the misalignment effects become more visible. Due to a complex combination of camera and object movement, the exposures cannot be aligned. However, if the second exposure is eliminated from the sequence the images can be aligned. In the resulting HDRI (c) the object are now better aligned, see (e). The clouds are still not perfectly aligned. This is due to a combination of an imperfect camera alignment and limited cloud movement.*

5.7.2 Movement removal using the variance detector

Figure 5.17 (a) shows a sequence of four exposures containing people walking through the viewing window. The background contains some moving objects too. These moving objects do not hamper the image alignment. After camera curve calculation, the exposures are merged into an HDRI H shown in



(a)

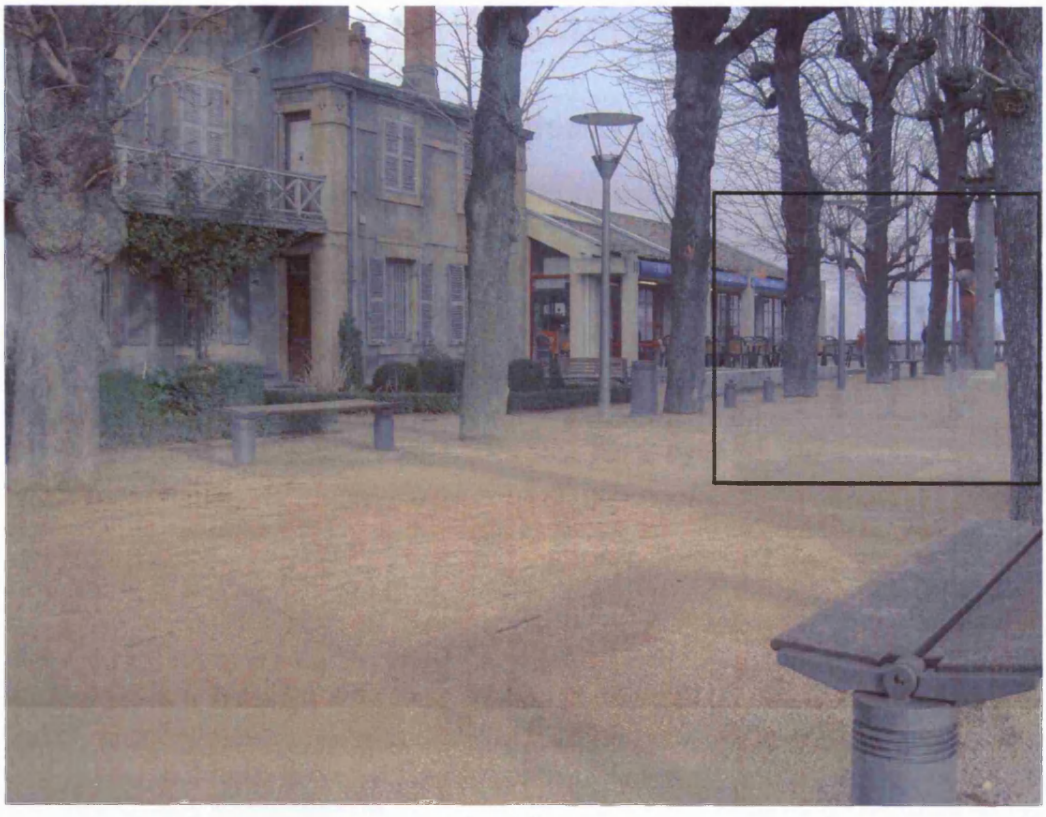
(b) (c)

Figure 5.17: *The set of exposures shown in (a) show several people walking passed the camera. Also in the background there are several moving people visible. The Variance Image VI (b) shows where the variance is the highest. Image segmentation results in the movement clusters shown in (c).*

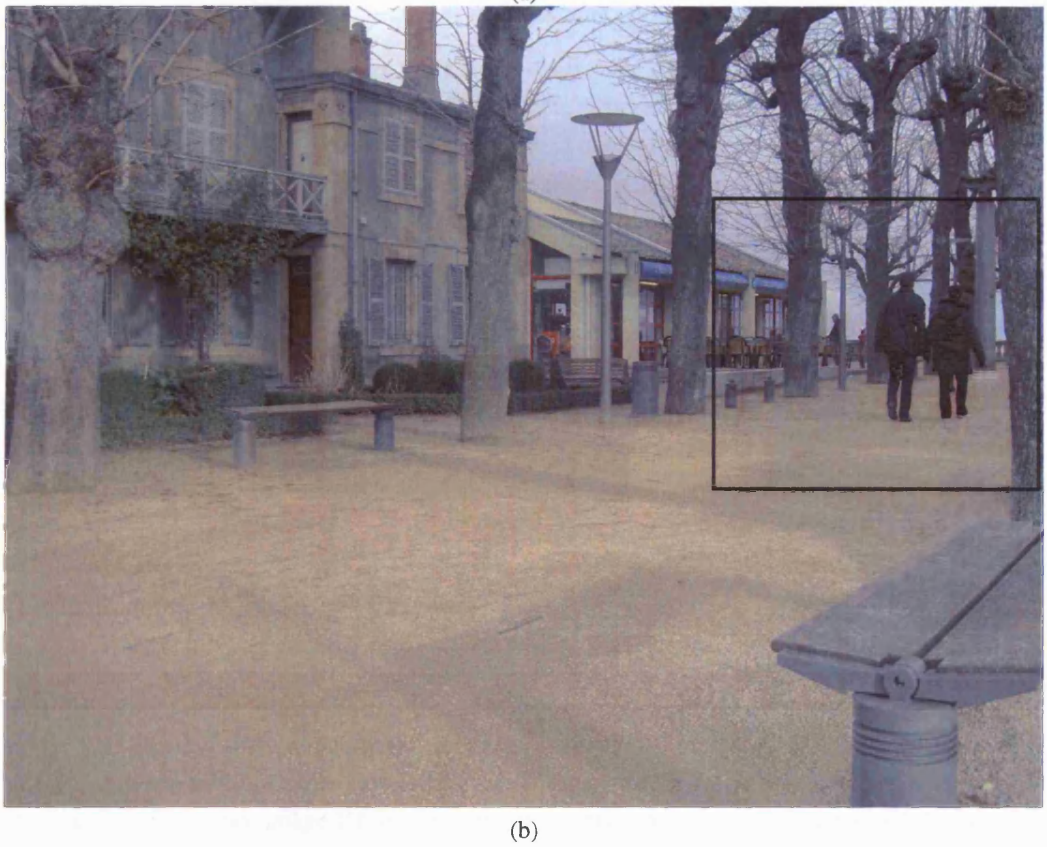
figure 5.18 (a). The moving people create a ghosting effect. This ghosting effect is not highly visible, but a close look at the irradiance inside the black square reveals the ghost-image of the lady with the red jacket, and (more vaguely) the other people in the scene.

The variance image VI created as explained in section 5.6.3 is shown in figure 5.17 (b). The brighter the intensity in VI, the higher the variance of that pixel is over the four exposures. The variance is high in the region affected by the moving people. Besides the moving people, several other bright variance pixels are revealed. The scene edges contain a high variance, but also other scene features. This is due to imperfections in the lens and camera sensor, visible at high contrast regions [RWPD05], but also due to the small misalignments that are still present after image alignment. These misalignments are expected, as the alignment method in section 5.5 only removes 2D translational and rotational misalignments. Applying a threshold (0.18) to VI and eroding and dilating this binary image VI_T results in a segmentation of the movement clusters, see figure 5.17 (c). For each of the identified clusters, an exposure is assigned that represents that cluster without saturation. Then the values in H are substituted using linear interpolation. The resulting HDRI $\overline{H}$ is shown in 5.18 (b). In this image $\overline{H}$ we see that the substitution is accurate, up to some minor artefacts, like the incomplete reconstruction of the man's leg in the figure.

Figures 5.19 and 5.20 present two other examples of movement removal. In figure 5.19 (a) the leaves inside the black square show considerable ghosting in the HDRI generated after image alignment. Using the variance image, this movement is effectively removed as can be seen in (b). Figure 5.20 (a) shows the HDRI generation after movement removal for the same exposures used to generate figure 5.14. There are several moving objects: the flag at the top of the ship's mast, the boat at the front right of the images which moves due to the movement of the water, and the people walking on the river bank. Using the variance image the movement clusters are effectively detected and substituted. Also, the movement of the water on the stone river bank is detected and substituted. Unfortunately not all rippling of the water



(a)



(b)

Figure 5.18: *(a) HDRI generated after image alignment. The exposures contain people walking past the viewing window. These do not obstruct the image alignment, but do create ghosting effect. (b) Movement removal using a variance image effectively substitutes the movement clusters.*

is detected, and some parts still show some blurring as a result of this. Figure 5.20 (b) and (c) show a close-up of the people on the riverbank, respectively before and after movement removal.

5.7.3 Movement removal using the uncertainty detector

Figure 5.21 illustrates how the creation of an uncertainty image UI can be used to derive the movement clusters for the same exposure sequence as shown in figure 5.17(a). The bright highlights in figure 5.21 (a) indicate uncertainty about the stability of the content of the pixels. The bright highlights correctly overlap with the position of the people that walk through the scene. Applying a threshold T_{UI} (= 0.7) means that it is assumed that if the uncertainty is larger than T_{UI}, there are most likely some misalignments due to the presence of moving objects. After applying erosion and dilation, well-defined movement clusters are created (b). Each cluster gets an exposure assigned, and the corresponding pixels in H are substituted by the irradiance values derived from the assigned exposure. The resulting HDRI $\overline{H}$ (c) is free from the artefacts created by the movement of the people walking through the scene. The variance movement detector shows similar results on this sequence of exposures as can be seen when comparing figure 5.21(c) with 5.18 (b).

To illustrate the potential of the local entropy images H_i, figure 5.22 shows the local entropy images for the first and second exposure shown in the exposure sequence of figure 5.17 (a). The entropy image has brighter values near edges or patterned surfaces. The entropy image has darker values when the intensity in the selection window W is uniform. Figure 5.22 (a) and (b) illustrate that, except for the areas showing the people, the two local entropy images are similar. Figure 5.22 (c) shows the absolute difference between the two local entropy images. As expected, this difference is high where the two exposures contain different scene objects, i.e. at the location of the people.

The uncertainty image shown in figure 5.21 (a) highlights a first potential problem associated with using the entropy measure as movement detector. Between the different exposures exists a transformation that is monotonic and increasing, where the middle part of transformation is usually injective. In theory this should not pose a problem when detecting movement using the equation given in equation 5.16. However, as already explained in section 5.6.4 the entropy is sensitive to saturation and under-exposures since it requires a injective function between the image values. Therefore, it is essential that the weights used to remove the influence from the saturation and under-exposure effectively remove any kind of saturation. Figure 5.21 (a) shows a high differential local entropy in areas affected by saturation effects (e.g., ground in front of the bench, trees, and shadows cast onto the house). This indicates that the weights used are not satisfactory. Until this point, we used the weighting scheme as defined by [MN99], which in fact depends on the camera curve. To remove the dependency on the camera curve, a new weighting scheme, the hat-function, is adopted. The weight associated with an intensity Z is given as:

$$w(Z) = \begin{cases} 1 & \text{if } T_L < Z < T_H \\ 0 & \text{elsewhere} \end{cases} \tag{5.19}$$

Suitable T_L and T_H are 0.15 and 0.85 respectively, for normalized intensity values $Z \in [0, 1]$.

This weighting scheme was applied to the same exposure sequence used to generate figure 5.19 (b). Figure 5.23 illustrate the uncertainty and variance images for the exposure sequence. The uncertainty image UI shown in (a) succeeds in detecting the movement of the leaves and branches in the front of the image. The variance image VI shown in (b) highlights similar movement areas as UI. However, while the computation for VI still uses the previous weight implementation, we can see differences between UI and VI in saturated and under-exposed areas, such as the highlight in the water. In UI

(a)

(b)

Figure 5.19: *(a) HDRI generated after image alignment: the leaves on the left hand side show movement. (b) Movement removal using a variance image effectively substitutes the movement clusters.*

(a)

(b) (c)

Figure 5.20: *(a) HDRI generated after image alignment and movement removal: the flag on the ship's mast, the boat at the front right, the water and the people on the river bank are effectively substituted. A close-up of the moving people before and after movement removal is shown in (b) and (c) respectively.*

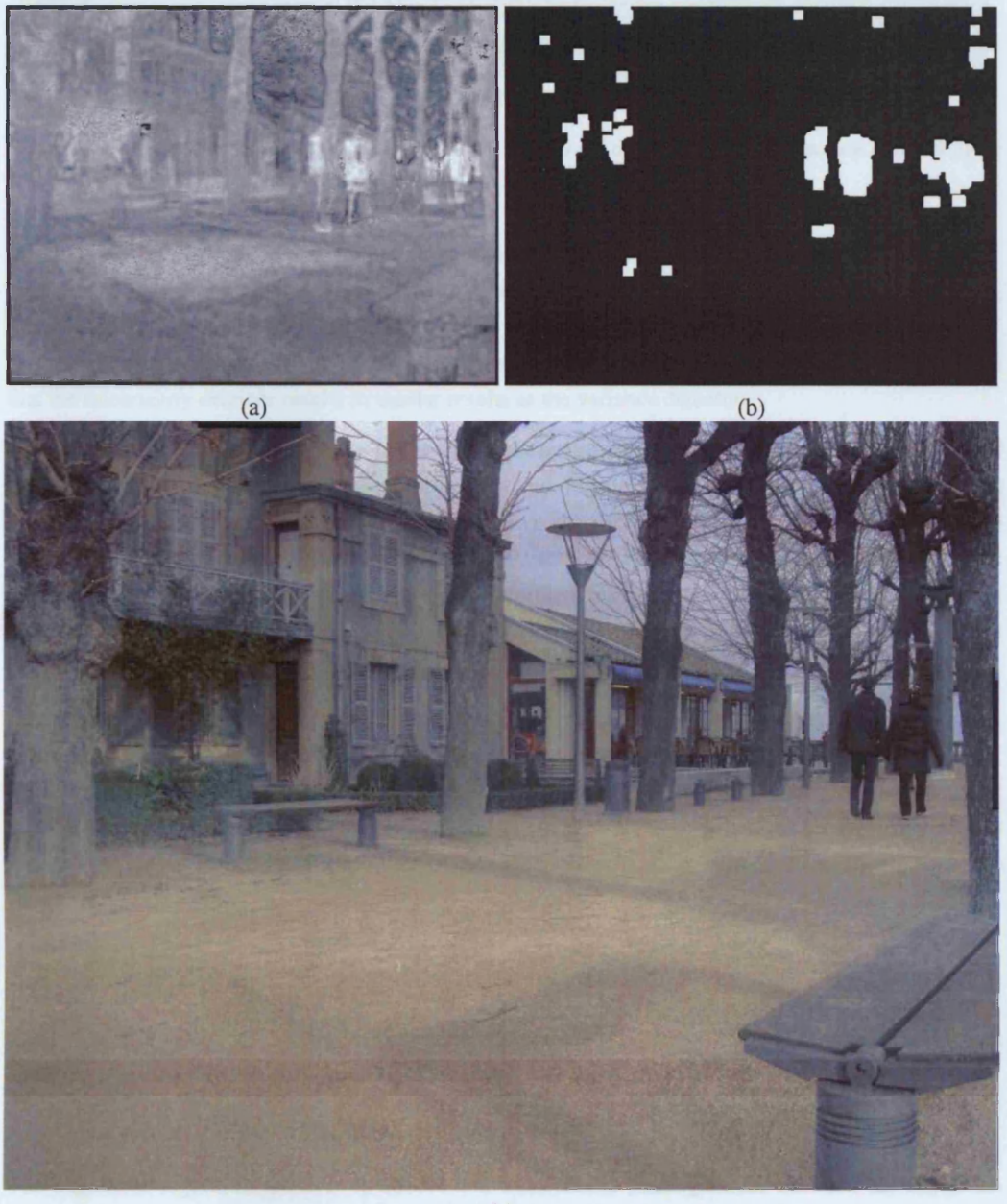


Figure 5.21: *(a) Uncertainty image for the same exposure sequence as in figure 5.17. (b) Binary uncertainty image UI_T after applying a threshold and two morphological operations. (c) Resulting HDRI: free from the artefacts created by the movement of the people walking through the scene.*

(a) (b) (c)

Figure 5.22: *(a) and (b) The local entropy images H_i of respectively the first and second exposure shown in figure 5.17 (a). (c) The absolute difference between (a) and (b).*

certain areas do not receive an uncertainty measure, and are left black. This means that no suitable exposure is available for those pixels that can be used to calculate an unbiased uncertainty measure, within the limits defined by the equation 5.19. The resulting HDRI $\overline{H}$ after applying a threshold to UI and movement cluster substitution is given in figure 5.23 (c). A comparison with figure 5.19 (b) shows that the uncertainty detector results in similar results as the variance detector.

The following example illustrates the power of the uncertainty image to detect movement regardless of the contrast between moving objects and the background. Figure 5.24 (a) shows three LDRIs from a set of five exposures; the tree branches and the leaves move throughout the capture. The variance image (c) detects the movement around the border of the tree correctly but fails to detect the movement that occurs inside the tree. The uncertainty image (b) detects movement inside the tree correctly, but fails to make a judgement about the sky due to too many saturation and under-exposure effects in that area. The HDRI before and after movement removal using the uncertainty image are shown in figure 5.25 (a) and (b).

The uncertainty detector is not perfect, and fails to detect movement in certain occasions. For instance, the rippling of the water in the exposure sequence used to generate the HDRI shown in figure 5.13, cannot be detected. The reason is that the rippling of the water is a local and small movement, smaller than the window W used to calculate the local entropy images H_i. As a result, the local entropy images are more or less the same for all exposures in the sequence at the water surface, resulting in movement detection failure.

5.8 Limitations and possible improvements

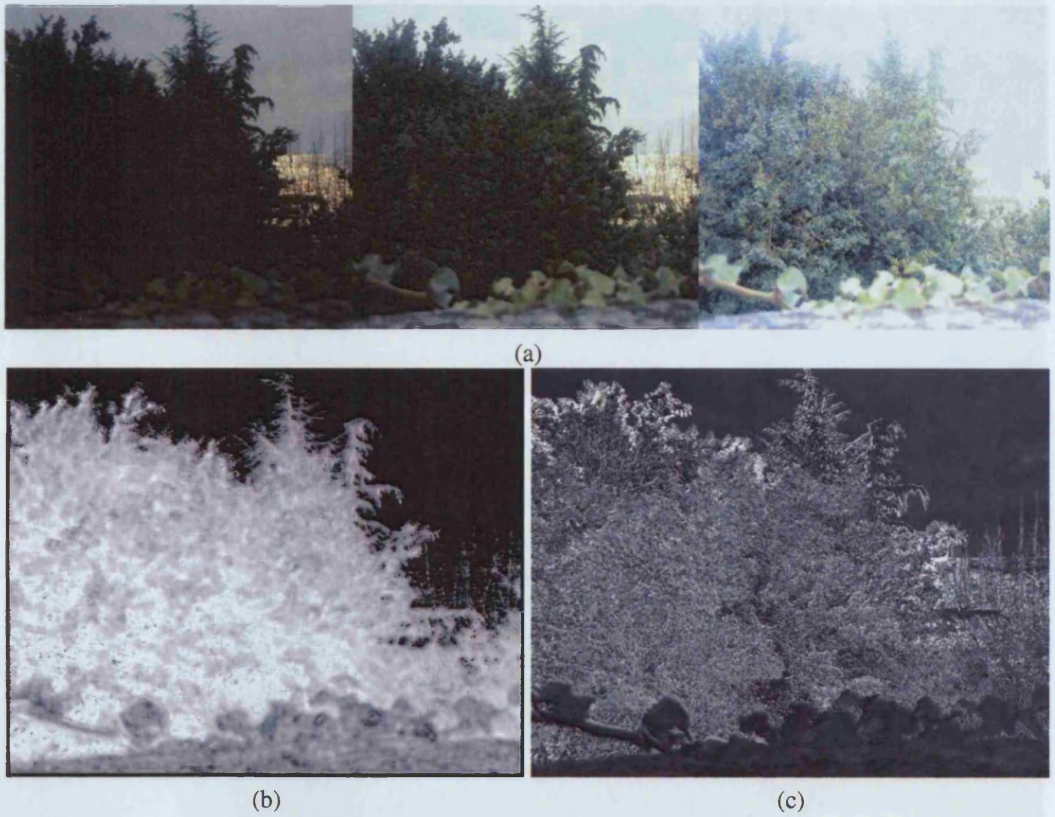
The following subsections discuss the limitations of the alignment method and object movement removal algorithm presented in this chapter. Several areas of future work are indicated.

A Alignment restriction to Euclidean transformation

The alignment algorithm retrieves rotational and translational misalignments between the image sequence. For most image sequences this limitation to a Euclidian alignment is satisfactory, however usually small misalignments remain. These small misalignments create high variance and high uncertainty values, and therefore both types of movement detectors presented are sensitive to these misalignments. The erosion operation, applied to the binary images VI_T and UI_T, is incorporated in the movement removal process to remove such small areas of high variance and uncertainty.

Nevertheless interesting future work would involve recovering the perspective transformation that exists between the different LDRIs to ensure an even better alignment. Applying this perspective transformation to align the LDRIs to the same viewpoint will introduce the problem that not all scene points visible

Figure 5.23: *(a) Uncertainty image. (b) Binary uncertainty image $U\,I_T$ after applying a threshold and two morphological operations. (c) Resulting HDRI: free from the artefacts created by the movement of leaves at the front.*



(a)

(b) (c)

Figure 5.24: *(a) Image exposure sequence: the branches and leaves in the tree move due to the wind. (b) Uncertainty image UI. (c) Variance Image VI. The resulting HDRI prior to movement removal is shown in 5.25 (a).*

(a)

(b)

Figure 5.25: *(a) HDRI H shows considerable ghosting effects. (b) HDRI $\overline{H}$ using UI shown in figure 5.24 (a).*

from this viewpoint are visible in all LDRIs. Such scene points will therefore receive a less reliable irradiance value.

B Robustness against large object movement

When a too large object (an object occupying a large area in the LDRI) moves in the scene, the presented alignment procedure may fail to align the different LDRIs. If the alignment had been successful, the camera curve calibration method, as currently implemented within the HDRI generation framework, would be unreliable as it would depend on a large portion of pixels that do not represent the same scene content.

The uncertainty measure, used to detect the movement pixels, does not require a-priori information about the camera curve. This offers the extra advantage that the detected movement clusters could be withheld throughout the camera curve calibration. The implementation of the camera curve calibration using this a-priori movement information has been left as future work.

C Non-rigid alignment

Some applications might want to capture large moving non-rigid objects, for instance when capturing the irradiance of a human body to map it onto an avatar as a texture. The person in the exposure can cause considerable non-rigid object movement, which can compromise the alignment. Even if alignment can be guaranteed, substituting a large moving object with irradiance values from one exposure, might return saturation and under-exposure, since the object (human body) will most likely occupy a large part in the viewing window. The result is that the final HDRI might not produce a more reliable higher dynamic range than the dynamic range present in one LDRI.

Finding a warping function between the exposures to align the human body, could be a solution to recombine the exposures into irradiance values. The methods from Kang et al [KUWS03] and Sand et al. [ST04a] made first steps into this direction, but as explained in section 5.3 these methods face problems, for instance, when aligning low-contrast areas. As a possible solution, the warping function could be estimated using mutual information [Vio95]. It should be noted however that the same problems apply as discussed in subsection A, as some scene points might not be visible in more than one LDRI.

D Lack of semantic information to interpret results

Both detection algorithms presented in this chapter, do not use any semantic information to define which pixels belong to a movement cluster or not. Since both movement detectors are sensitive to small mis-alignments and image noise, morphological operations are required to detect the pixels that are most likely subject to movement. However, these morphological operations are not always 100% successful.

To overcome such problems, the movement detectors could be implemented such that they search for certain objects that might have caused movement. For instance, when capturing a public space, the moving object is most likely a person. Therefore a general human shape could be used to filter the images VI and UI to improve the chances of detecting the correct movement clusters.

Such a condition would require limited user-interaction as it would allow the user to select a pre-defined movement shape. Depending on the application for which the HDRIs are used, such limited user interaction might not be too disturbing.

E Substitution by one exposure

The substitution of the movement clusters by the most suitable exposure relies on the fact that this exposure does not show saturation or under-exposure in the area extended by the movement cluster. If

saturation and under-exposure effects are present the substitution results in incorrect irradiance values. Also, in such an event the substitution will most likely be visible, for instance if the saturation and under-exposure effects appear at the borders of the movement cluster. This is because during the weighting of the exposures into H those corrupted irradiance values were probably suppressed, and therefore unlikely to have been highly visible in H. At the border of the movement cluster this might introduce unwanted artefacts.

An improvement to the presented algorithm would be to detect if several, instead of one, exposures exist that do not show movement in the movement cluster. A combination of these exposures could then be used to substitute the movement cluster, in a similar way as defined in equation 2.32 in chapter 2.

Another problem associated with the substitution of the movement clusters by one exposure per movement cluster, is that some moving objects might become present twice. This occurs, for instance, when an objects moves rapidly through the exposure sequences, and appearing in two movement clusters.

F Computation time Uncertainty Image

Computationally expensive histogram calculations are needed to calculate the uncertainty image UI. Whereas the computation of VI is less expensive. The computation time required to calculate UI ranges around several seconds, while the computation of VI occurs in near real-time. The computation time for UI depends on the resolution of the LDRIs, the window w used to calculate the local entropy, and the number of bins M used to calculate the histogram. It might still be favoured to use VI for the movement detection when timing is an issue, and if the camera curve is accurately known and there is no LCM present.

G Choice of thresholds for VI and UI

Assigning the thresholds to detect movement can be difficult. Both VI and UI provide a measure of the likelihood that movement is present in a certain pixel, but both images can also be corrupted by side-effects. For the variance detector these are: an incorrect camera curve, incorrect exposure times or aperture widths, incorrect image alignment, an incorrect weighting scheme, and image noise. For the uncertainty image these are: incorrect image alignment, an incorrect weighting scheme, and image noise.

The morphological operations will ensure only large movement clusters are detected, which enhances the chance that a true moving object is detected. Nevertheless the thresholds defined in section 5.6 to create VI_T and UI_T will most likely not be suitable for every exposure sequence to detect the correct movement cluster.

H Illumination changes

Illumination changes occurring during the exposure capture result in erroneous irradiance values with the current methodology of HDRI generation. Illumination changes might be less likely to occur during the exposure capture, nevertheless it remains an interesting area for future work.

Illumination changes can be detected using, for instance, a lightmeter, but a more interesting method would detect the illumination changes from the exposures instead of using an external device. A possible automatic illumination change detection algorithm would use the information from the shadows in an image. A crude algorithm would state that if the direct illumination changes, the non-shadow areas are more affected by these illumination changes than the shadowed areas. The illumination change can therefore be detected based on the variance, or uncertainty that exists in non-shadows areas when the shadowed areas have a low variance or uncertainty.

5.9 Chapter summary

HDRIs differ from conventional LDRIs in that they represent the scene content without saturation or under-exposure. HDRIs are usually created using multiple exposures, which are images captured with varying exposure settings and from the same viewpoint. The time lapse between first and last exposure increases the likelihood that camera and object movement occurs. Both effects create blurring and ghosting effects in the final HDRI. As a result, HDRI capture nowadays occurs in inflexible circumstances: the camera is fixed on a tripod, and object movement is avoided. HDRIs can be captured with an HDRI camera too, but these cameras are expensive, and are equally flawed by their lengthy capture time.

Limited research has been carried out to remove the errors in HDRI generation due to the camera and object movement. In this thesis, an analysis of the current state of the art is made, and improvements are offered for detected flaws. The resulting developed method allows the exposures to be captured with a hand-held camera, and allows limited object movement.

An alignment method, available in the literature, approximates the camera movement by a translation in horizontal and vertical direction. In this chapter we improved the alignment by representing the misalignment as a euclidean transformation: translation and rotation. The results showed that recovering rotational misalignments improved the quality of the HDRI. However, it was also indicated that recovering perspective transformations would most likely further improve the results.

To the best of our knowledge, only one object movement algorithm for exposures exists. The algorithm detects the movement using a variance measure, and substitutes the irradiance in the HDRI in the movement area by irradiance values derived from one exposure. The variance measure is the variance of a pixel's irradiance in the different exposures, high variant pixels are usually corrupted by movement. The drawback of this method is that the variance detector relies on the camera curve, which is often estimated from the same set of exposures. This poses a chicken-egg problem: the incorrect exposures introduce errors in the camera curve, which in turn reduces the performance to detect the movement in the exposures. Also, the variance detector relies on a threshold to detect pixels with a high variance. This means that low contrast movement is less likely to be detected. A solution was offered in this chapter by detecting the movement in the exposures using an uncertainty measure: the higher the uncertainty, the more likely movement has corrupted the pixel's irradiance value. The resulting movement detection algorithm succeeds in detecting movement across images for which the intensity values are linked by an injective function. This makes it usable for detecting movement in exposure sequences. The uncertainty measure does not require the knowledge of the camera curve, and therefore pixels corrupted by movement can now even be withheld during the camera curve calibration.

The limitations and possible future work were indicated in section 5.8. The main limitations are that if a moving object is too large alignment might fail, that the substitution of the irradiance values by values from one exposure can be error-prone, and the requirement of having an injective function between the image values of the exposures. Also, the uncertainty detector requires considerably more computation time. Besides camera and object movement, it is possible that during the LDRI capture the scene illumination changes, for instance due to cloud movement. This has a significant impact on the HDRI generation and so far no solutions have been proposed to take care of these illumination changes.

Chapter 6

Radiance capture of dynamically lit scenes

6.1 Introduction

Most illumination methods require a capture process that involves taking photographs of the scene in order to extract its geometry and original illumination parameters (radiance and reflectance values of scene surfaces). This capture process is often long, tedious and error-prone. Therefore, it is usually carried out under highly controlled conditions, requiring, for instance, that the original illumination conditions are fixed, known, or easily measurable.

Inverse illumination proceeds by applying the inverse of the radiance equation on the geometry and scene radiance, to retrieve the reflectance properties of the scene. The radiance of the scene is usually extracted from many different images, captured from different viewpoints. For the inverse illumination equations to make sense, the radiance in the scene needs to be known for all points under the same global illumination settings. This was already explained in chapter 2, where in section 2.3 it was illustrated how the lighting effects in the scene radiance are detected based on the position of the light sources in the scene, and the scene geometry. Section 2.4 explained how the radiance of all scene points needs to be known under the same global illumination settings for inverse illumination.

Capturing the radiance of all scene points under the same illumination is not as straightforward as it seems. Illumination changes are more common in real-world scenes than one might think. The most common example of dynamic illumination is due to cloud movement. Large object movements, like the movement of tree branches in the wind, can cause a similar effect. An indoor scene can be subject to similar dynamic lighting effects when outdoor light falls through a window onto the scene. People walking in a scene (but not necessarily visible from the camera's viewpoint) can block light in an irregular, undesired manner. Unfortunately, the longer it takes to capture a scene, the more likely these scene illumination changes take place. Therefore, the larger and more complicated a scene, the more likely illumination changes occur.

Most illumination methods available in the literature ignore illumination changes and present results showing controlled, usually indoor, scenes with static illumination. This chapter, however, presents a new radiance capture methodology for relighting applications of scenes subject to dynamic illumination. The methodology allows the user to capture different parts of the scene at different times of the day and/or under different types of illumination conditions. The radiance is reconstructed from originally uncalibrated photographs of reflective spheres. A novel registration method is presented that allows a semi-automatic calibration of the position of the reflective sphere. Furthermore, the radiance values in the image of the reflective sphere are *back-projected* onto the scene geometry in a fast and mathematically accurate manner, removing distortion and pinching effects often introduced when using reflective

spheres. The back-projected textures create a *coherent* radiance distribution, that is captured under the same illumination conditions, per reflective sphere. The obtained coherent radiance distribution is used to estimate the reflectance values of the surfaces positioned near the reflective sphere. Once the reflectance properties are known, the scene can be relit using a novel illumination pattern.

The presented methodology is designed for relighting applications. Nevertheless, several aspects of the methodology can be used in other types of illumination methods. Lightprobes are often used in common illumination to extract the scene light sources or to simulate the scene illumination [ALCS03]. Part of the presented radiance capture methodology can be used to ensure the lightprobe images are treated correctly in common illumination, as distortion effects are also a problem in such types of illumination methods.

This chapter is organized as follows. Section 6.2 analyses in more detail the illumination changes that can occur, and discusses the errors that exist when using reflective spheres to capture the scene radiance. Section 6.3 discusses the methodology presented in this chapter. The methodology consists of 4 different steps: radiance capture (section 6.4), calibration (section 6.5), radiance registration (section 6.6), and inverse illumination (section 6.7). Section 6.8 discusses how this is integrated in an actual relighting application, and shows results of two scenes relit using the presented method. In section 6.9 an overview of the limitations of the presented methods and some directions for future improvements are given. Finally, section 6.10 gives a summary of the contributions and methods of this chapter.

The algorithms developed and presented in this chapter have been presented as a poster at Eurographics Rendering Workshop and have been listed in a technical report at UCL [JNVL06a]. The relighting method is under submission for publication in IEEE Transactions on Visualisation and Computer Graphics (TVCG). The software package reflect, developed for this purpose, has been implemented and designed with the help from Msc. students Anders H. Nielsen and Jeppe Vesterbæk. The synthetic scene, used in section 6.8.1, has been designed by Anders H. Nielsen and Jeppe Vesterbæk.

6.2 Dynamic illumination: a source for problems

This section should convince the reader that illumination changes occur frequently in a real environment, and that this dynamic behaviour is not limited to outdoor scenes, but is also present in indoor scenes. This section is organized as follows. Section 6.2.1 gives some examples of types of dynamic illumination and illustrates how this affects the radiance/texture extraction. Section 6.2.2 explains how a coherent radiance distribution can be retrieved using images of a reflective sphere, how these images are usually treated and what problems this invokes. Section 6.2.3 briefly discusses some related existing illumination methods and describes the contributions of this chapter to the state of the art in illumination methods for mixed reality.

6.2.1 Illumination changes do happen

In its simplest form, inverse illumination extracts the material property of a scene point p from the scene geometry, the radiance of p, and the radiance of all other scene points visible from p. With the geometry and scene radiance known the radiance equation given in equation 2.18 can be constructed, and subsequently the material property extracted as it is the only remaining unknown parameter. For diffuse materials, the equations can even be further simplified: the reflectance of a point p is defined as $f(p, \omega_i, \omega_r) = f(p)$. The reflectance can then be calculated as the ratio of the radiance of p divided by the irradiance of p. The inverse illumination procedure, as outlined here, only guarantees a correct

material property extraction if the available radiance distribution is *coherent*, that is, resulting from the same global illumination settings.

Extracting a coherent scene radiance distribution is not straightforward. As a result, the illumination methods in the literature avoid such situations. While these methods obtain high-quality results on static illuminated scenes, it is unlikely that these methods will produce the same results on dynamically lit scenery. Indeed, only very few researches address this issue [Deb04][TA05]. Nevertheless, dynamic scene illumination is more common than static illumination. Uncontrollable scenes, such as outdoor environments and public places, are often subject to illumination changes. Cloud movement can result in (sometimes very rapid) illumination changes. Pedestrians reflect or block the light in an unpredictable manner. Large object movements, like trees in windy environments, can result in light perturbations. Light falling into a room through a window can create unexpected, but not necessarily negligible, illumination changes.

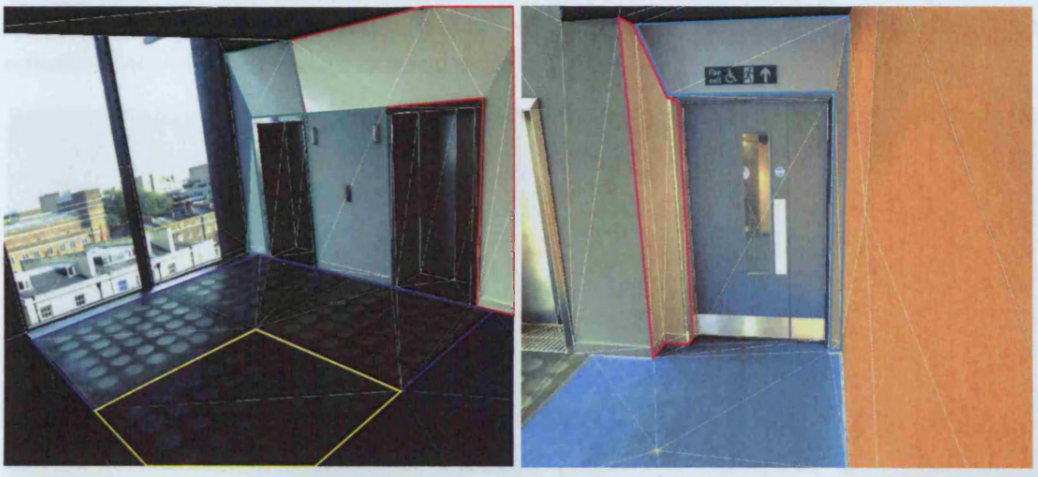
When the scene radiance is extracted from a set of HDRIs captured from different viewpoints, the lengthy HDRI capture can further increase the possibility of illumination changes, especially for outdoor scenes, since the sun moves over time. The radiance is often projected onto the geometry using reconstruction software which extracts the radiance as a texture. Capturing a large scene from different viewpoints will rarely be successful after a first set of images are captured. For instance due to carelessness, some parts of the scene are not visible in any of the images captured, which therefore requires a retake of the scene to retrieve the missing radiance data. Unless the scene is a controlled indoor scene, recapturing a scene to extract missing radiance nearly always occurs under different illumination conditions.

An example is given in figure 6.1, which shows two screen shots of the same reconstructed scene, captured under dynamic illumination. The polygons (wireframe) that lie inside the same contour (red, green, purple, yellow, pink or turquoise) have their textures extracted from the same image. This means that the polygons inside one contour have a coherent radiance distribution. The radiance distribution within different coloured contours are not necessarily coherent. As an example, the image used to extract the radiance of the polygons inside the yellow contour is captured under a different global illumination than the image used to extract the radiance of the polygons inside the purple contour. The light through the window changed during the day, due to the clouds in the sky and the change in the position of the sun. The illumination in the building shown cannot be switched off with a switch, but is triggered by movement on each floor. Therefore, the light coming through the translucent tiles of the floor and the ceiling is triggered by people walking on the floors below and above the floor reconstructed in this example. Some parts of the scene had to be recaptured a few times, resulting in images captured at different times of the day. All this together gave the scene capture a very dynamic character.

Another example is given in figure 6.2 which shows two images of the same scene. The two images have been captured with identical exposure settings but with a time interval of eleven minutes. This time lapse was sufficient to change considerably the scene illumination. While the sun casts clear shadows in (a), its intensity has been significantly reduced in (b).

6.2.2 Extracting scene radiance from one HDRI

In the previous section, we discussed how extracting the radiance from multiple images can result in radiance incoherency. Thus it makes sense to capture the entire scene radiance with only one image. Although it will be nearly impossible to capture the entire scene radiance with only one image (due to occlusion effects), it is possible to get an approximation of the scene radiance using one image of a reflective, specular sphere. Such an image is called a *lightprobe image* and the reflective sphere used



(a) (b)

Figure 6.1: *(a) and (b) show two screen shots of the VRML model of an indoor scene, affected by the illumination coming through the window and through translucent ceiling and floor tiles. The light coming through the ceiling and floor tiles is uncontrollable because the light switches on the floors directly above and under the reconstructed scene are triggered by people walking on these floors. It took almost a full day to capture the entire scene: the HDRI capture process was lengthy, and the scene had to be captured in different sequences, as it emerged during the reconstruction that additional images were required. The result is a textured 3D model, with an incoherent radiance distribution. In (a), the texture inside the contours in red, green, purple and yellow, are incoherent due to a different global illumination setting. The same is true for the texture inside the pink and turquoise contours in (b). The texture inside the pink contour shows severe colour bleeding from the orange wall. The texture inside the turquoise contour shows almost no colour bleeding. While this might be partially due to the orientation of the surface towards the orange wall, it is mainly due to the fact that more light was reflected by the orange wall at the time of capture of the texture inside the pink contour, than at the time of capture of the texture inside the turquoise contour.*

(a) (b)

Figure 6.2: *(a) and (b) are captured with the same exposure settings (aperture: f2.8, exposure speed: $\frac{1}{500}$). (a) is captured on a winter day at 15h15, (b) is captured eleven minutes later at 15h26. The time lapse of eleven minutes is sufficient to create a totally different global illumination. While the sun casts clear shadows in (a), its intensity has significantly been reduced in (b).*

in such context is often referred to as a *lightprobe*. An example of a lightprobe image positioned on the floor is given in figure 6.3 (a) and from a different viewpoint in (b). The lightprobe image contains 3D omni-directional scene information; it shows radiance from behind, the front and the sides of the reflective sphere (all relative to the camera position).

(a) (b)

Figure 6.3: *(a) A reflective sphere positioned on the floor, and in front of the door. (b) A close-up image of the reflective sphere, which is called a lightprobe image. The lightprobe image shows information about the scene from behind, from the sides, and from the front of the sphere (all relatively to the camera position).*

The following subsections discuss how it is possible to capture 3D omni-directional information using a lightprobe, and provide some further terminology that is used throughout this chapter.

A Lightprobe images

Figure 6.4 (a) illustrates how the content in a lightprobe image needs to be interpreted using a 2D simplified representation of the lightprobe-camera set-up. The sphere is positioned in the centre of the camera viewing window. The sphere is specular, so a ray through a pixel of the camera is reflected around the normal at the reflection point, in the same angle as the incoming angle. When the camera is positioned at distance D from the centre S of the reflective sphere, two tangent rays can be constructed that are reflected at 90° and, eventually, intersect with the scene at p_1 and p_2. In 3D, all tangent points would form a circle on the sphere. The surface created by the sphere and the tangent lines forms a cone-like volume. All points inside this volume are obscured from the camera by the sphere; this is illustrated (in 2D) by the grey area in (a). Figure 6.4 (b) illustrates the situation when the camera is positioned at $D \rightarrow +\infty$, the tangent rays are parallel to the viewing direction. The obscured volume behind the sphere is now a cylinder (again displayed in grey), with the same width as the diameter of the sphere. If the radius R of the sphere is negligible relative to the size of the scene, the obscured cylindric volume is negligible too. In that case, the entire 3D scene, visible from the origin of the sphere, is captured in one image (while ignoring possible occlusion effects).

In reality it is impossible to position the camera at infinity, nor to produce an infinitesimally small reflective sphere. In addition, the smaller the sphere, the lower is the resolution at which the radiance

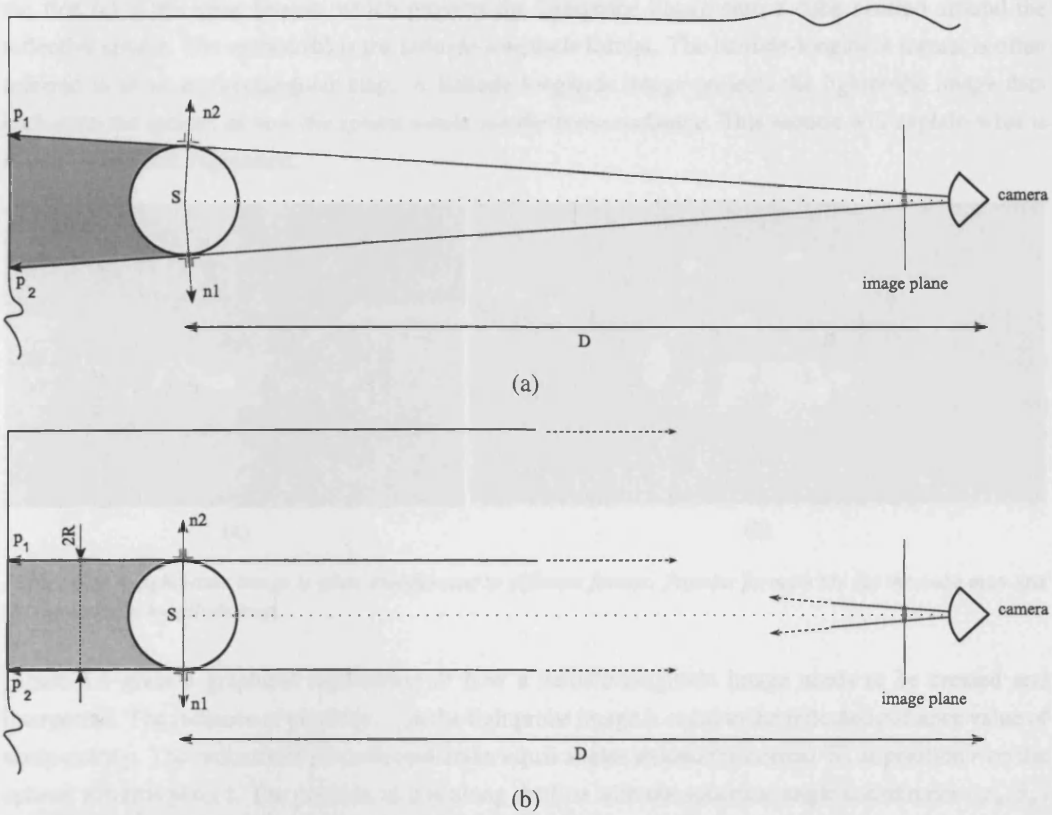
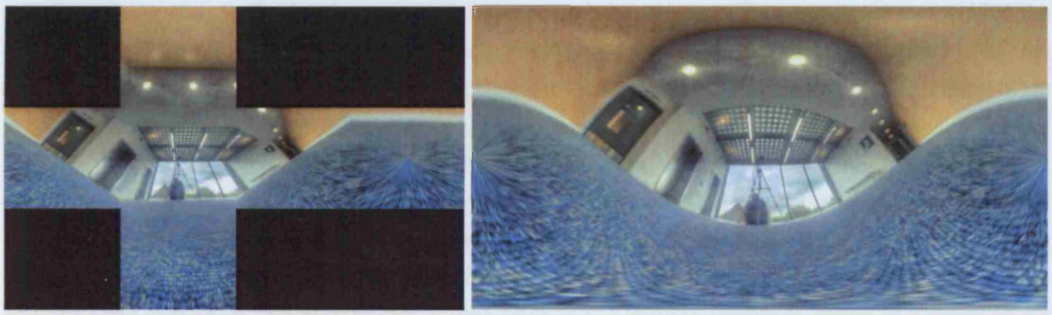


Figure 6.4: *Image (a) and (b) show a 2D interpretation of how lightprobe images can capture omni-directional information. A ray from the camera hits the reflective sphere and is reflected around the normal $\vec{n}$ at the reflection point, into the opposite direction but with the same angle, due to the specular behaviour of the reflective sphere. In (a), the two tangent rays are reflected at $\vec{n_1}$ and $\vec{n_2}$ at 90° and intersect with the scene at p_1 and p_2. The grey cone-like region behind the sphere is not visible from the camera's viewpoint. (b) Suppose that the camera is positioned infinitely far from the sphere ($D \rightarrow +\infty$). The tangent lines are now parallel to the viewing direction, and intersect with the scene in p_1 and p_2. This time the invisible region (grey) has the shape of a cylinder. The size of this cylinder depends on the dimension of the sphere. If the sphere is infinitesimally small ($R \rightarrow 0$) the invisible volume converges to 0 and the entire 3D environment, at least those points visible from the origin of the sphere, would be projected onto the lightprobe image, albeit with zero resolution.*

is captured. Therefore, when capturing the scene radiance, a trade-off needs to be made between the resolution and the size of the invisible volume behind the sphere. It should be noted, that the combination of two lightprobe images from two different directions can result in a true omni-directional radiance capture. Nevertheless this option is not considered in this chapter, as it would require two different lightprobe captures, which can result in illumination differences between the two captures.

B Latitude-Longitude image: pinching and distortion

A lightprobe image is often transformed into a different format. Figure 6.5 shows two popular formats, the first (a) is the cube format, which projects the lightprobe image onto a cube centred around the reflective sphere. The second (b) is the *latitude-longitude* format. The latitude-longitude format is often referred to as an equirectangular map. A latitude-longitude image projects the lightprobe image data back onto the sphere, *as how the sphere would see the scene radiance*. This section will explain what is meant by this last expression.



(a) (b)

Figure 6.5: *A lightprobe image is often transformed to different format. Popular formats are (a) the cube map and (b) the latitude-longitude map.*

Figure 6.6 gives a graphical explanation of how a latitude-longitude image needs to be created and interpreted. The radiance of pixel $t(k, l)$ in the lightprobe image is equal to the reflected radiance value of scene point p. The radiance of p is reflected under equal angles around the normal $\overrightarrow{n}$, at position r on the sphere, towards pixel t. The position of p is along the line with the spherical angle coordinates (φ_p, θ_p) in the local coordinate system $X'Y'Z'$ positioned in the centre of the sphere. The line that connects the centre of the sphere S with p intersects the sphere at point q with spherical coordinates (R, φ_q, θ_q) = (R, φ_p, θ_p). The transformation from lightprobe image to latitude-longitude image *rearranges* the radiance values of the pixels in the lightprobe image according to the spherical angles (azimuth and elevation) of their corresponding scene points, each can also be interpreted as the projection point q of p on the sphere. In other words, the radiance of point p is stored at (φ_p, θ_p) in the latitude-longitude image. If the size of the latitude-longitude image is $N \times M$ then $M = 2 \times N$, the azimuthal angle covers M values from 0 to 2π at intervals of $\frac{2\pi}{M} = \frac{\pi}{N}$; the elevation angle covers N values from 0 to π at the same interval as the horizontal axis.

Note that in the previous discussion it is implicitly assumed that the sphere is positioned in the centre of the camera's viewing window. The viewing direction of the camera is equal to the axis X' of the local coordinate system positioned at the centre of the sphere and therefore the axes Y' and Z' are parallel to the camera's image plane.

According to this transformation method, the transformation from a lightprobe image to the latitude-longitude format requires information about the geometry of the scene. With the dimension of a reflective sphere being small compared to the dimensions of the scene, it is often assumed that the scene lies

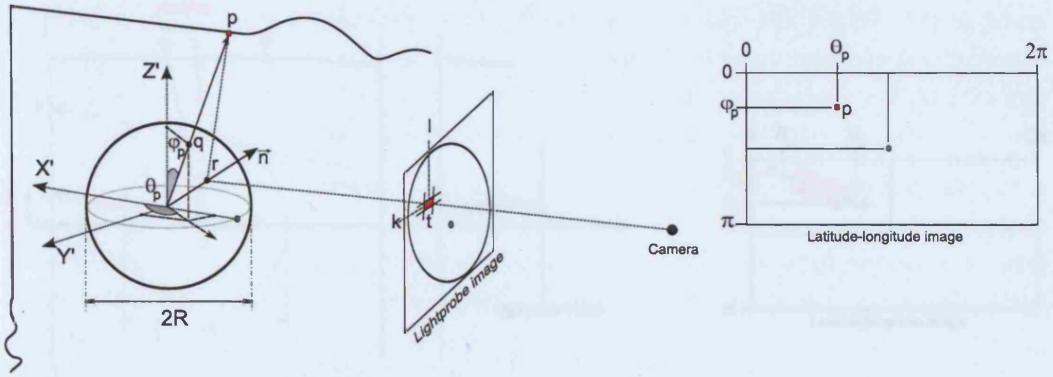


Figure 6.6: *The transformation from lightprobe image to latitude-longitude image rearranges the pixel values according to the spherical angles (azimuth and elevation) of their corresponding scene points.*

at infinity compared to the local set-up of the sphere and the camera. This assumption simplifies the transformation from lightprobe image to latitude-longitude image. A 2D graphical interpretation of how the latter assumption influences the transformation is given in figure 6.7. Instead of storing the radiance value of pixel t at the spherical coordinates of point p, it is stored at the spherical coordinates of the line Sq_b. The line Sq_b is simply a line parallel to the line rp and is constructed from the angle that the line rt makes with the normal $\vec{n}$. The angle that the line Sq_b makes with X' is therefore equal to $\pi - (\delta + \alpha)$. The line Sq_b and rp are parallel and intersect at infinity, and in the case of a truly infinite scene, they would intersect at p. In reality however, this assumption implies that the radiance value of p is actually *re-positioned* to point b. We refer to this as *distortion* in the latitude-longitude image as a result of the incorrect assumption of having an infinitely large 3D scene.

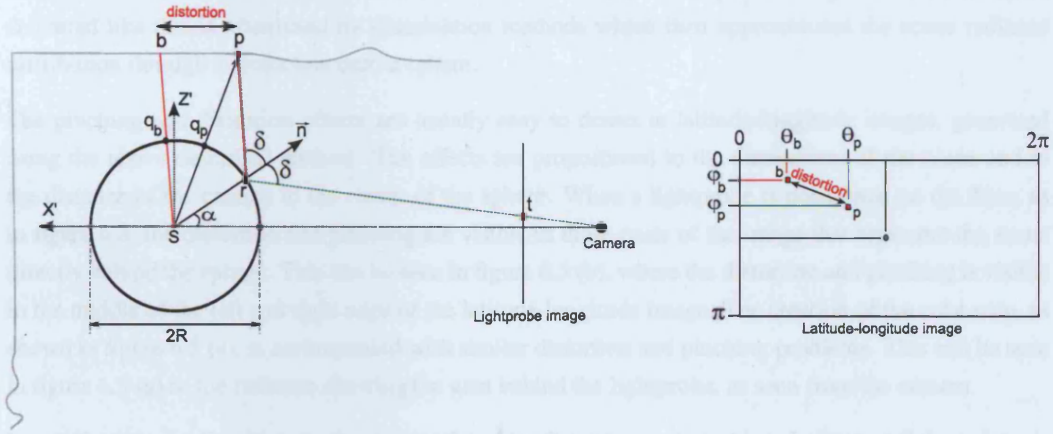


Figure 6.7: *When assuming that the scene lies at infinity, the spherical coordinates of point p can be approximated by the spherical coordinates of the ray through S and parallel to the line rp (both shown in red). In reality this misplaces the radiance of point p to the point b.*

Transforming the lightprobe image to the latitude-longitude format under the above defined assumption still requires the knowledge of the local set-up of the camera and the sphere: the distance D of the lightprobe to the centre of the camera, and the physical dimension R of the sphere. Again, abstraction can be made of knowing these parameters as is illustrated in figure 6.8.

The tangent line from the camera to the sphere, shown in figure 6.8, should intersect with scene point k and return the radiance from k. The distortion, introduced by assuming the set-up defined in figure 6.7, would distort the radiance of point k to point l. Ignoring the distance from the camera compared to the

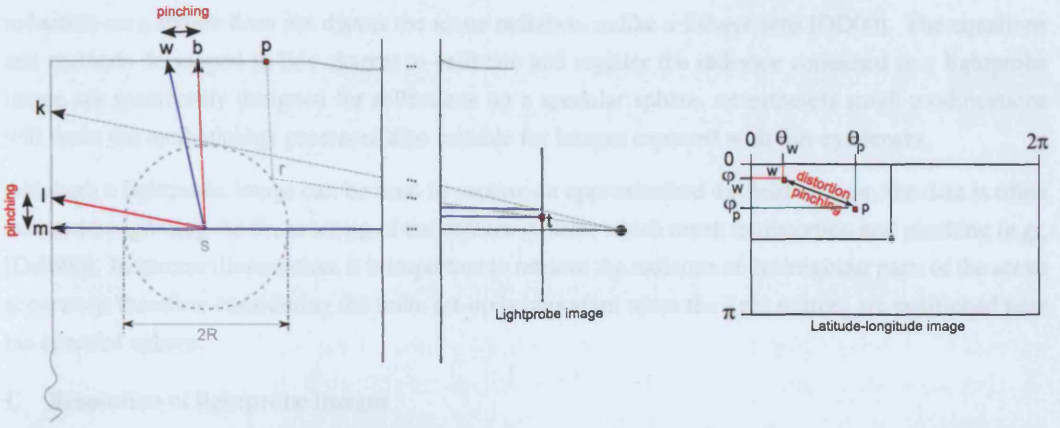


Figure 6.8: *When the distance between the camera and the sphere is infinitely large ($D \rightarrow +\infty$) and the radius of the sphere is infinitesimally small ($R \rightarrow 0$), the transformation from lightprobe image to latitude-longitude image becomes very simple. In reality, the line tangent to the sphere (grey line) intersects with scene point k. The method described in figure 6.7 introduces distortion and actually re-positions the radiance value of k to scene point l (red line). The aforementioned assumptions ($D \rightarrow +\infty$ and $R \rightarrow 0$) further reduce the transformation to an orthogonal projection which would re-position the radiance from k to scene point m (blue line). When the same procedure is applied to the radiance of scene point p, this radiance will be re-positioned to scene point w. The error introduced by re-arranging point b to w is called pinching.*

centre of the sphere can be approximated as an orthogonal projection. Also assuming that the sphere is infinitesimally small, would assign the radiance along the tangent ray to that of scene point m in figure 6.8. The error introduced by skewing scene point l to m is defined as *pinching*. Software packages like HDRshop [HDR] and Photomatix[Pho] apply the transformation as discussed in this paragraph, as they do not have any notion of the 3D scene, nor of the local camera-sphere set-up. The transformation delivered like this is often used by illumination methods which then approximates the scene radiance distribution through a projection onto a sphere.

The pinching and distortion effects are usually easy to detect in latitude-longitude images, generated using the above described method. The effects are proportional to the dimensions of the scene and to the distance of the camera to the centre of the sphere. When a lightprobe is positioned on the floor, as in figure 6.3, the distortion and pinching are visible in those parts of the image that represent the scene directly behind the sphere. This can be seen in figure 6.5 (b), where the distortion and pinching is visible in the middle of the left and right edge of the latitude-longitude image. The creation of the cube map, as shown in figure 6.5 (a), is accompanied with similar distortion and pinching problems. This can be seen in figure 6.5 (a) in the radiance showing the area behind the lightprobe, as seen from the camera.

It is difficult to predict the error that is introduced by distortion and pinching. In figure 6.8 the point p is distorted and pinched to point w. The distance between p and w depends on the geometry of the scene. If the surface on which p, b and w lie is perpendicular to the line Sp, the distortion is less than if this surface is parallel to that same line. A numerical analysis of the distortion based on an example is given in section 6.6.3.

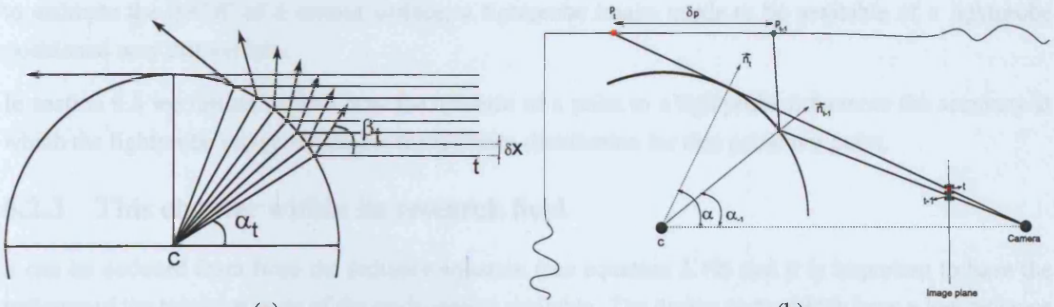
Another way to capture omni-directional information is by using a fisheye lens. The combination of two images captured with a fisheye lens could return the same information as a lightprobe image captured with a reflective sphere. For inverse illumination applications it often suffices to have information about the hemisphere above a certain point, in that case only one image would be sufficient. There are two reasons why we did not use a fisheye lens for our application. The first is that it is much cheaper to use a mirror ball than a fisheye lens. The second reason is that when ignoring fresnel refraction, the

reflection on a sphere does not distort the scene radiance, unlike a fisheye lens [OD00]. The equations and methods developed in this chapter to calibrate and register the radiance contained in a lightprobe image are specifically designed for reflections on a specular sphere, nevertheless small modifications will make the methodology presented also suitable for images captured with fish-eye lenses.

Although a lightprobe image can be used to capture an approximated 4π field of view, the data is often misused by ignoring the finite set-up of the capture system, which result in distortion and pinching (e.g., [Deb98]). In inverse illumination, it is important to retrieve the radiance of the brightest parts of the scene accurately, therefore considering the finite set-up is important when the light sources are positioned near the specular sphere.

C Resolution of lightprobe images

A lightprobe image does not represent the entire scene with the same resolution. The part of the scene lying along but opposite the viewing direction of the camera is captured at the highest resolution. The further the scene point deviates from the line connecting the camera with the sphere, the poorer the resolution. Figure 6.3 (b) clearly shows how a large part of the scene is squeezed into the outer circle of the lightprobe image. When capturing lightprobe images, the photographer needs to take this into account by orienting the camera such that the most important parts of the scene are captured at a higher resolution if possible. The non-uniform sampling pattern is further illustrated in figure 6.9 (a).



(a) (b)

Figure 6.9: *(a) Sampling the scene using a lightprobe image results in a non-uniform sampling pattern. The rays that hit the sphere at equidistant intervals are projected under non-uniform angles. The angles are larger for rays through the pixels near the border of the sphere. (b) The ray through pixel t is tangent to the sphere and is reflected towards scene point p_t. The ray through pixel $t-1$ is reflected towards scene point p_{t-1}. The distance δp gives a measure of the spatial sampling distance in the 3D scene, obtained with the lightprobe image.*

To simplify the analysis in this section it is assumed that the sphere is captured using an orthogonal projection, as illustrated in 6.9 (a). The rays hit the sphere at equidistant intervals of size δx. Consider a ray through pixel t, the angle of the normal through the reflection point of this ray is α_t:

$$\alpha_t = asin(\frac{|t| \cdot \delta x}{R}) \tag{6.1}$$

with R the radius of the sphere, and $|t| < \frac{R}{\delta x}$. The derivative of α_t to the parameter t is given as:

$$\frac{d\alpha_t}{dt} = \lim_{dt \to 0} \frac{\alpha_t - \alpha_{t+dt}}{t - (t+dt)} = \frac{1}{\sqrt{1 - \frac{t^2 \cdot \delta x^2}{R^2}}} \cdot \frac{\delta x}{R} > 0 \tag{6.2}$$

Since δx, t and R are always positive, the quantity $d\alpha_t$ is positive. Equation 6.2 shows that as t grows, $d\alpha_t$ increases. The angle β_t is equal to $2\alpha_t$, when considering an orthogonal projection, therefore the scene is sampled non-uniformly. Considering a perspective projection instead of orthogonal boosts the

non-uniformity even more. This is also discussed in [TI04].

To analyse the largest spatial sampling interval, the distance between point p_t and p_{t-1} can be calculated from the intersection of the reflected rays through pixel t and $t-1$ and the 3D scene, with t being the tangent pixel. This is illustrated in figure 6.9 (b). Although it is not guaranteed that the distance δp is the largest sampling distance, as it depends on the geometry of the scene, it is a good indicator of the sampling resolution for a particular scene.

D A lightprobe image provides a radiance distribution for points near the lightprobe

A lightprobe image reflects the scene radiance onto the camera sensor. Each pixel represents the radiance $L(p_i, \omega_i)$ for which ω_i is the angle that the ray $p_i r$ forms with the surface normal of the scene point p_i, where the notations of figure 6.6 are used. If the point p_i is specular, $L(p_i, \omega_i)$ is not constant as ω_i varies. In other words, the radiance captured with the lightprobe image is dependent on the position of the lightprobe image. Also, occlusion effects are different for different positions of the lightprobes.

Thus the radiance captured with a lightprobe only provides a good scene radiance estimate for the points positioned near the lightprobe. In other words, the points near the reflective scene *see* a similar radiance distribution to the one captured in the lightprobe image. This knowledge influences our presented capture process considerably, see section 6.4. When using a lightprobe image to reconstruct the scene radiance, this will only provide a good radiance estimate for the points near the lightprobe used. Therefore, to estimate the BRDF of a certain surface, a lightprobe image needs to be available of a lightprobe positioned near that surface.

In section 6.8 we further explain how the distance of a point to a lightprobe influences the accuracy at which the lightprobe image represents the radiance distribution for that particular point.

6.2.3 This chapter within its research field

It can be deduced from from the radiance equation (see equation 2.19) that it is important to have the radiance of the brightest parts of the environment available. The darker parts, which have a low radiance value, are masked by the brighter parts due to the integration. Therefore some methods simplify the radiance capture by measuring the light source intensity and its direction and position [BG01]. This is not practical when considering larger scenes with multiple large light sources when the illumination changes.

Some existing methods use a lightprobe image to capture the overall radiance of the scene, acquired from photographs of a reflective sphere [Deb98][ALCS03] or a fisheye lens [SSI99]. These methods using reflective spheres, assume the sphere's position, or at least the position of the camera used to capture it, is known. These positions are measured by hand or using calibration boards. Existing methods using reflective spheres to model the incoming light, in particular methods that use environment mapping, do this in an incorrect and/or inefficient manner. Often it is assumed incorrectly that the sphere reflects a 4π field of view. This is less important when the light comes from a distant part of the scene than when it comes from nearby surfaces. In this chapter a novel back-projection method is present that derives the radiance of scene points from the radiance values in the lightprobe images without expensive ray-tracing calculations but using simple OpenGL commands.

The illumination methods often split the scene in two parts, a local and a distant part (e.g., [Deb98]). The radiance of the distant scene is usually represented with a lightprobe image. However, when the local scene is large, one such lightprobe does not represent the radiance correctly for all scene points. The BRDF for scene points far from the lightprobe will be less accurate. In this chapter this problem is

resolved by capturing many lightprobe images for lightprobes near the surfaces for which the BRDF is calculated, and applying a coherency principle on the BRDF values that compensates for the inaccuracies due to the distance to the lightprobe. This allows capturing different parts of the scene using different input images captured under a different illumination setting.

Capturing many lightprobe images introduces the problem of calibrating the lightprobes, which, when carried out through calibration boards, hampers and delays the capture process. The method presented in this chapter removes the need to calibrate the lightprobe images at capture time and calibrates the lightprobes in a post-processing step using a novel semi-automatic calibration procedure. This improves the state of the art in illumination capture considerably, as it eases the capture process.

Some recent work addresses the issues of dynamic illumination [TA05] [Deb04]. In [TA05] an alternative for the radiance capture is proposed. The geometric model of an outdoor scene comes from scanned data [Lei] onto which radiance textures are mapped. These textures are extracted from photographs taken at different times of the day so when stitched together, differences are clearly visible in the overlapping areas. Based on estimated visibility and shadow maps, radiance values are normalized to create new textures with consistent illumination. The result is a homogeneous relighting over the complete model. Although very interesting, the method has some drawbacks as several assumptions are made: radiance values are homogeneous inside a shadow, the sun is fully visible (not partially covered by clouds), its position can be estimated, and the contribution from the sky is not directly considered. As a result, this method is limited to certain types of relighting, and inverse illumination would not be possible from this scene capture. The focus of the method presented in this chapter is similar to that of Debevec et al. [Deb04]. However, although the methodology presented here aims at achieving similar results, the approach of Debevec et al. is still very different. The BRDF of the materials in the scene is extracted iteratively, starting from a representative BRDF sample for each material in the scene. This BRDF sample is reconstructed by capturing the material under controlled illumination (in their example at nighttime). Debevec et al. also use reflective spheres, but do not treat them for distortion. The positions of the spheres are estimated using two calibrated cameras. Finally, their results focus on outdoor scenes with the main light source at infinity. As explained in the remainder of this chapter, our approach provides a general method to capture the scene radiance, allowing relighting of scenes lit under different uncontrollable illumination conditions with less restrictions and assumptions than existing methods.

6.3 Methodology

Figure 6.10 gives an overview of the methodology presented in this chapter in the context of a relighting application. The entire methodology, from scene capture to relighting, consists of five different modules, which are split in two different groups. The first group of tasks cover the reconstruction of the scene in the widest sense: from geometry capture to radiance registration. The second group deals with the reflectometry, or the reflectance calculation. The five modules are briefly discussed here:

- **Scene capture**: this step involves the image capture from which the scene radiance and scene geometry is extracted. The image capture requires a specific procedure to deal with the illumination changes, as is explained in section 6.4. In short, the image capture involves capturing two sets of images. A first set consists of input images used for the geometry and texture extraction. The second set consists of lightprobe images, where each lightprobe image is associated with one input image. The lightprobe images are used to reconstruct the scene radiance at the time of capturing the associated input image.

- **Lightprobe calibration**: the positions of the reflective spheres used to capture the above mentioned lightprobe images do not need to be known a-priori, but are calculated based on a minimum of 6 pairs of corresponding points in the 3D scene and the lightprobe image using a novel calibration method presented in section 6.5.

- **Radiance registration**: the radiance from a lightprobe image is back-projected onto the scene, to reconstruct the radiance distribution present at the time of capturing the associated input image. As explained in a previous section, an image of a reflective sphere fails to capture a complete 4π field of view which is often ignored by methods using reflective spheres to capture the scene radiance. In section 6.6 a novel back-projection procedure is provided that correctly describes the relation between the scene points and pixels in the lightprobe image.

- **Inverse Illumination**: based on the input data gathered with the previous steps, the reflectance values of the materials in the 3D scene are estimated. In this chapter, an example of diffuse BRDF extraction is presented, but a similar strategy to [BG01] could be used to estimate a more sophisticated model that includes specular effects such as the model presented by Ward et al. [War92]. The BRDF estimation is an iterative process controlled by the difference between a rendered image and a reference image. The influence of the distance between a lightprobe and a scene point and the BRDF calculation of that scene point is explained and solved through a novel coherency principle. More details are given in section 6.7.

- **Relighting**: once the reflectance values and geometry are known, the scene can be relit with a different illumination setting. This is discussed in section 6.8.

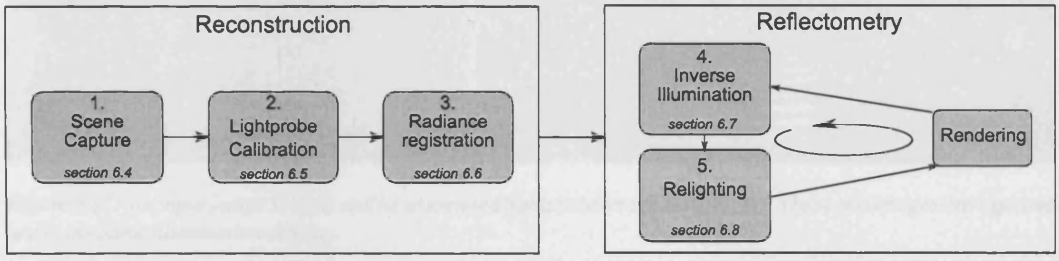


Figure 6.10: *The relighting methodology consists of five different steps split into two groups: reconstruction and reflectometry. The first group cover the scene reconstruction from geometry to radiance. The second groups deals with the actual reflectance calculation.*

The sub-modules of the methodology presented in figure 6.10 can be used in a different context than relighting. The principle of capturing a lightprobe image for each input image can be useful for any type of illumination method hindered by illumination changes throughout the capture. The radiance calibration and registration can be used in any illumination method that requires accurate knowledge about the scene illumination. The semi-automatic calibration provides an easy-to-use method to find the position of the light sources in the scene, based on one lightprobe image. In that context the radiance capture as presented in this chapter can be used in common illumination, relighting and physically-based illumination.

6.4 Scene capture: set-up

The scene radiance and geometry are captured as follows. Two sets of photographs are taken and converted to HDRIs:

- The first set contains N photographs of the scene, called *input images* I_i, with $i \in [1..N]$. The input images I_i are used for the geometric reconstruction and to steer the reflectance calculations. The input images are calibrated using reconstruction software (e.g., ImageModeler). Surfaces for which the reflectance needs to be calculated have to be visible in at least one input image. The input image is used as a reference image during the reflectometry stage.

- The second set contains N photographs of a reflective sphere positioned at various places in the scene, called *lightprobe images* LP_i, with $i \in [1..N]$. The lightprobe images LP_i are used for the lighting extraction during the inverse illumination and relighting steps. The different positions of the sphere do not need to be known during the image acquisition, because they are calibrated in a subsequent step.

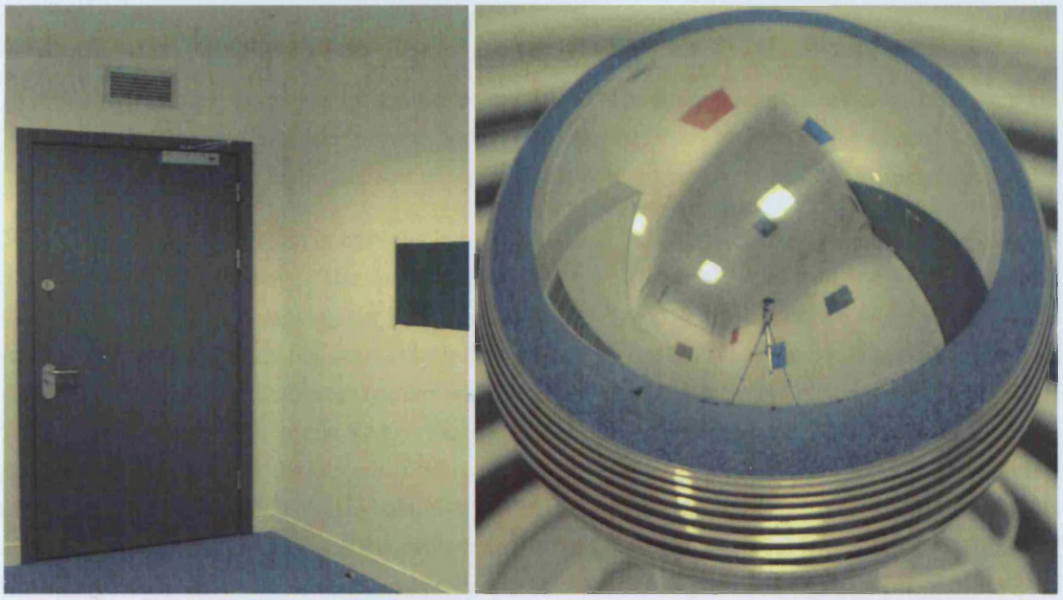


Figure 6.11: *An input image I_i (left) and its associated lightprobe image LP_i (right). These two images are captured under the same illumination settings.*

Every input image I_i has one lightprobe image LP_i assigned; the set $\{I_i, LP_i\}$ needs to be captured under the same illumination conditions. However, for $i \neq j$, $\{I_i, LP_i\}$ and $\{I_j, LP_j\}$ can be captured under different illumination conditions. The position of LP_i needs to be close to the surfaces visible in I_i, or at least it should be positioned near to the surfaces for which the reflectance is estimated using the input image I_i. An example of an image pair $\{I_i, LP_i\}$ is presented in figure 6.11.

To enable the calibration of the reflective sphere within the 3D scene, it is important to know the ratio between the radius of the sphere and the dimensions of the scene, as is discussed in section 6.5. This ratio is easy to obtain by measuring a reference object in the scene and the diameter of the sphere.

The reconstruction of the 3D geometry of the scene is carried out with 3D reconstruction software. ImageModeler [Rea] was used in this thesis, but any other reconstruction software is suitable. The reconstruction process approximates the scene with a triangulated mesh, and extracts the textures from the input images I_i. During the extraction process the following classification takes place:

- Triangles sharing the same material are grouped into *material clusters* MC_i, with $i \in [1 : M]$, where M is the total number of different materials in the scene.

- Triangles belonging to the same material cluster i, and for which the radiance texture is extracted from the same input image j, are grouped into an *illumination cluster* IC_{ij}, with $j \in [1, N]$, where N is the number of input images. Each illumination cluster has one texture assigned that contains the radiance information retrieved from input image j.
- An MC that shows a patterned texture is labelled as being *patterned.* Patterned in this context means that the material does not have an homogeneous appearance, and local material differences are present. This information is used in section 6.7 during the reflectometry process. Patterned MCs receive a per-pixel calculated BRDF, while non-patterned MCs receive one global BRDF.
- Polygons belonging to a material for which no reflectance will be calculated are grouped into a *neutral cluster* N_{ij} with $i = M + 1$ and $j = N + 1$. During the inverse illumination calculations, described in section 6.7, the neutral cluster will act as a textured area light source. The texture of this area light source is extracted from the lightprobe images through the back-projection process described in section 6.6. To achieve the back-projection, the neutral geometry should have a dummy texture assigned with a non-zero resolution. The actual radiance values in the texture are not important at this stage. It is important however that each triangle of the neutral cluster has texture coordinates associated that describe uniquely that triangle in the dummy texture.

This hierarchical organization of triangles into IC's and MC's is illustrated in figure 6.12, which shows a 3D model of a real scene. The size of the door in the 3D model is set to its physical dimension ($2.17m$) in order to correctly register the size of the lightprobe ($\varnothing 6.35cm$) relative to the reconstructed model. The patches inside the same coloured contour are from the same material cluster, depicted as $MC_1, MC_2, \ldots, MC_6$. Different contours within one MC form illumination clusters, denoted as IC_{11}, $IC_{12}, \ldots, IC_{19}$ for MC_1 in the figure. The carpet (MC_2) and the door (MC_3) are labelled as being patterned. The material of the latter does not really contain a patterned textured, but does contains several un-modelled details (doorknob, keyhole) that give it a patterned appearance. To preserve these details after the relighting, a per-pixel BRDF estimate is made, instead of one global BRDF for the MC under consideration. This is further discussed in section 6.7. The ceiling is considered as being neutral, no reflectance will be calculated for the material of the ceiling. The neutral cluster does not necessarily need to consist only of the surfaces on which the light sources lie. In the example shown the walls could also have been considered neutral geometry. Appendix D explains how the grouping of the polygons into ICs and MCs can be carried out when using ImageModeler to model the scene.

The requirement that I_i and LP_i are captured under the same illumination settings, might sound contradictory with the problem statement outlined in this chapter. Usually the scene illumination is static enough to ensure that two subsequent HDRIs are captured under the same illumination settings. Nevertheless if this cannot be ensured, a solution is to capture the lightprobe image and the input image simultaneously. This would require a trade-off between the portion of the scene that is visible in the input image and the resolution of the lightprobe image. Another solution is to extract the entire radiance from the lightprobe images, and to use these lightprobe images as reference images during the reflectometry. However, if possible, it should be avoided to capture the radiance from the lightprobe images only, as this would reduce the resolution at which the scene radiance is known, which might hamper the reflectometry calculations.

6.5 Lightprobe Calibration

It was mentioned that the position of the sphere used to capture a lightprobe image does not need to be known at acquisition time. The positions of the sphere and of the camera used to capture the sphere are

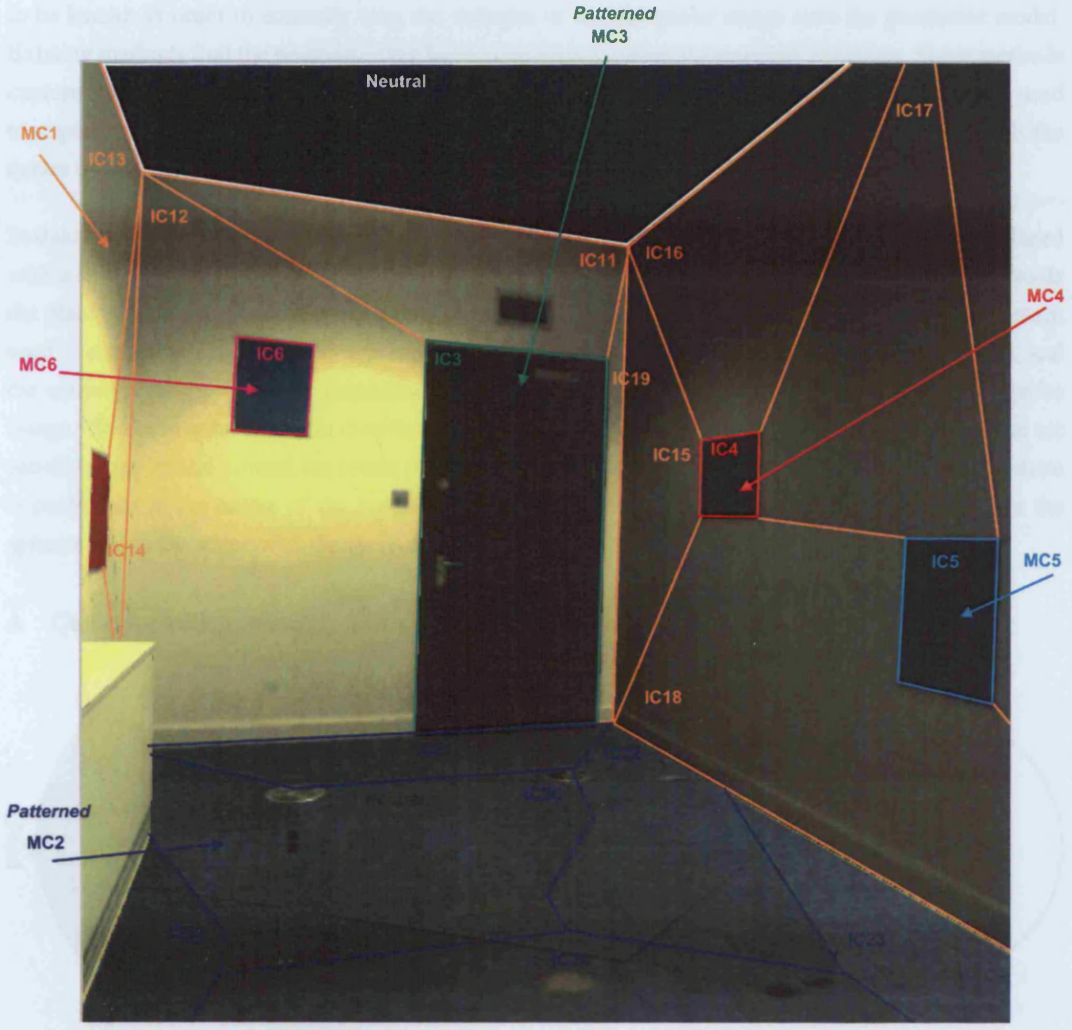


Figure 6.12: *Scene composition: the parts of the scene with the same material properties are grouped into a material cluster (areas with the same contour colour and labelled MC_i, with i an MC's index number). The parts of the scene visible in the same input image and with the same material properties are grouped into an illumination cluster (areas labelled IC_{ij} with i the IC's index into MC_i or $IC_{ij} \in MC_i$). There is no texture assigned to the ceiling in this example, therefore it is considered as being neutral, and no reflectance calculation will proceed on this surface. The door and the carpet are considered to be patterned.*

estimated using a novel calibration procedure, see section 6.5.1. The reflective spheres used are typically not purely specular. Section 6.5.2 discusses how the reflectance of the reflective sphere needs to be calibrated.

6.5.1 Position calibration

The position of the sphere and the position of the camera used to capture the lightprobe image need to be known in order to correctly map the radiance in the lightprobe image onto the geometric model. Existing methods find the position of the lightprobe by measuring the required positions. Some methods capture the sphere twice, from two different viewpoints, and only measure or calibrate the cameras used to capture the two images[Deb04][MDA02][LKG+03]. The intersection of the two rays through the centre of the two cameras returns the position of the lightprobe.

In this thesis all the efforts to calibrate the camera and sphere prior to the capture is reduced and replaced with a simple procedure to calculate the position of the lightprobe after the capture. This method avoids the placing of fiducials in the scene to calibrate the cameras and lightprobe. The position of the camera used to capture the lightprobe image is estimated based on the dimensions of the sphere and scene, and the scene geometry. The only requirement is that the sphere is captured in the centre of the lightprobe image. This is to reduce camera distortions (lens distortion, aberrations, optical vignetting, etc.) that are usually symmetrical around the centre of the image. No special efforts were made to ensure the sphere is positioned in the centre of the image. A visual verification proofed to be sufficient to calibrate the spheres inside the scene with the desired precision.

A Camera position relatively to the position of the sphere

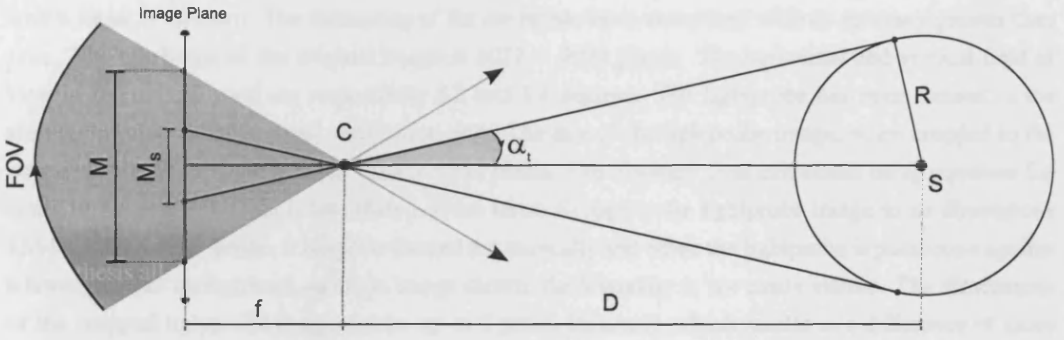


Figure 6.13: *Notations used to calculate the distance D of the camera C from the reflective sphere centred in S.*

The distance D of the camera (C) to the centre of the reflective sphere (S) can be estimated if the internal parameters (such as the vertical field of view (fov) of the lens and the total vertical resolution (M) of the camera sensor), the physical dimension (R) of the sphere and the pixel resolution (M_s) of the projection of the reflective sphere in the lightprobe image are known. A simple procedure to estimate the camera position is by assuming that the camera operates like a pinhole camera as is depicted in figure 6.13. Based on the assumption that the sphere is centred in the field of view of the camera, the distance of the camera from the centre of the sphere can be approximated using simple trigonometry. The tangent of the angle $\frac{fov}{2}$ is given as:

$$tg(\frac{fov}{2}) = \frac{\frac{M}{2}}{f} \tag{6.3}$$

with f the focal length of the camera. The tangent of the tangent line to the sphere is given as:

$$tg(\alpha_t) = \frac{\frac{M_s}{2}}{f} \tag{6.4}$$

The combination of equation 6.3 and 6.4 gives:

$$tg(\alpha_t) = \frac{M_s}{M} tg(\frac{fov}{2}) \tag{6.5}$$

Knowing α_t we can calculate the distance of the camera from the centre of the sphere:

$$D = \frac{R}{sin(\alpha_t)} \tag{6.6}$$

With the distance D known, we can express the position of the camera as $[-D, 0, 0]$ in the local coordinate system $X'Y'Z'$ of the sphere (depicted in figure 6.6).

The calculations as outlined here, assume a pinhole camera model while in reality the camera will be subject to lens distortions, aberrations, glare, and vignetting effects that compromise the accuracy of the presented calculations. To improve this approximation, the camera's internal distortion could be measured in advance by using a calibration board [Zha00]. The specular sphere used is usually a ballbearing and these exist in all sorts of precision. Calculating the influence of an imperfect sphere on the calculations is left as future work. In this chapter, as is explained in the results in section 6.8, the calibration as outlined here operates at an accuracy that is higher than the accuracy in the obtained geometric reconstruction. Therefore, the influence on the calibration of the imperfection of the sphere has been neglected in this chapter.

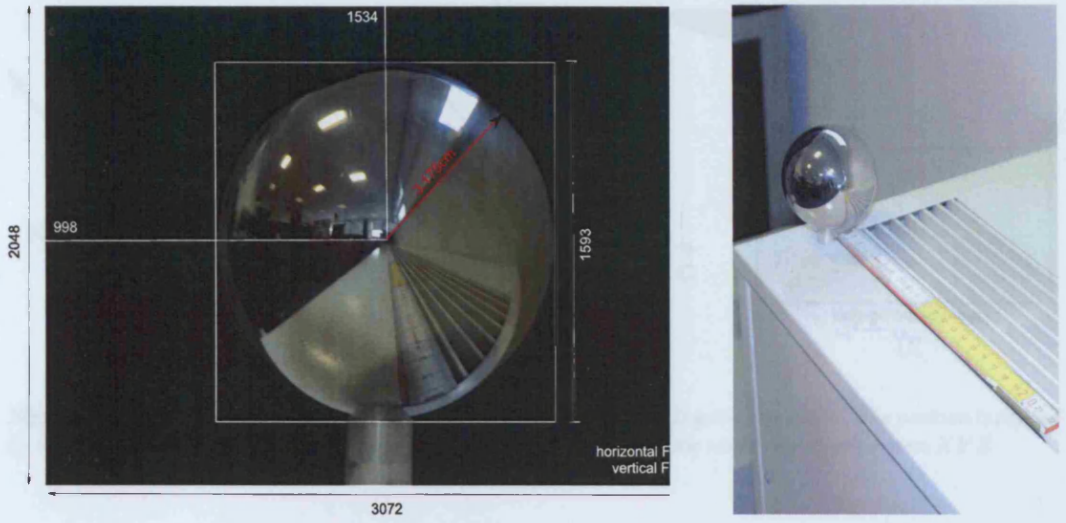
As a test case, see figure 6.14, we placed a lightprobe at a distance of $\approx 155.2cm$ from the camera lens[1] with a focus of $400mm$. The measuring of the set-up has been completed with an accuracy greater than $1cm$. The resolution of the original image is 3072 $\times$ 2024 pixels. The horizontal and vertical field of view of the camera used are respectively 5.2 and 3.4 degrees. The lightprobe has been centred in the viewing window using a visual verification only. The size of the lightprobe image, when cropped to the bounding box of the lightprobe is 1554 $\times$ 1554 pixels. The distance D as calculated using equation 6.6 result in $D = 153.6$. This is less than $1.57cm$ error. Cropping the lightprobe image to its dimensions 1554×1554 is error-prone, it has to be carried out manually and when the lightprobe is positioned against a homogeneous background, as in the image shown, the boundary is not easily visible. The dimensions of the cropped lightprobe image can be up to 5 pixels incorrect, which results in a difference of more than $1cm$ on D.

B Absolute position of the sphere in the world coordinate system

To find the absolute position of the sphere in the world coordinate system XYZ it is assumed that the (incorrect) conditions of figure 6.8 apply:

- The scene lies at infinity compared to the dimensions of the sphere.
- The radius of the sphere is infinitely small: $R \rightarrow 0$.
- The distance between the sphere and the camera is infinite: $D \rightarrow \infty$

[1] Sigma AF 135-400mm f/4.5-5.6 APO Aspherical RF



(a) (b)

Figure 6.14: *Measuring the distance of the camera to the lightprobe. (a) The dimensions of the image and lightprobe are indicated. (b) The position of the lightprobe and camera have been measured manually with a ruler. The measurements are not exact, a measurement error of more than* $1cm$ *is expected.*

It was already discussed that with these assumptions it is possible to find the position of the points visible in the lightprobe image relatively to the sphere, albeit in an incorrect, distorted manner. Although the radius of the sphere and the distance between the camera and the sphere are known they are ignored in the derivation of the absolute position of the sphere within the 3D scene for reasons of simplicity. Using the above described conditions the lightprobe image LP can be transformed to the latitude-longitude map LL using software packages such as HDRshop [HDR]. The relation between the latitude-longitude image and the scene set-up, under the above defined assumption, is visualized in figure 6.15.

The problem of finding the position and orientation of the reflective sphere relatively to the 3D scene can be reformulated as the problem of finding the transformation that maps the world coordinate system XYZ to the local coordinate system $X'Y'Z'$, as is illustrated in figure 6.15. The transformation can be written as an affine transformation W, a translation followed by a rotation and a scaling:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = W \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{6.7}$$

$$= \begin{bmatrix} a & b & c & T_x \\ d & e & f & T_y \\ g & h & i & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{6.8}$$

The vector $T = [T_x, T_y, T_z]$ defines the translation vector from the origin of XYZ to the centre of the sphere (see figure 6.15). The parameters a through i are defined by the relative rotation and scaling between XYZ and $X'Y'Z'$. In total there are 9 unknowns in equation 6.8 (3 rotation angles, 3 scaling numbers, and the translation vector), where the coefficients a through i are non-linearly related. For reasons of simplicity we assume that the coefficient are not related, in that case there are in total 12 unknowns in equation 6.8. One set of corresponding points $(x, y, z, 1)$ and $(x', y', z', 1)$ between XYZ

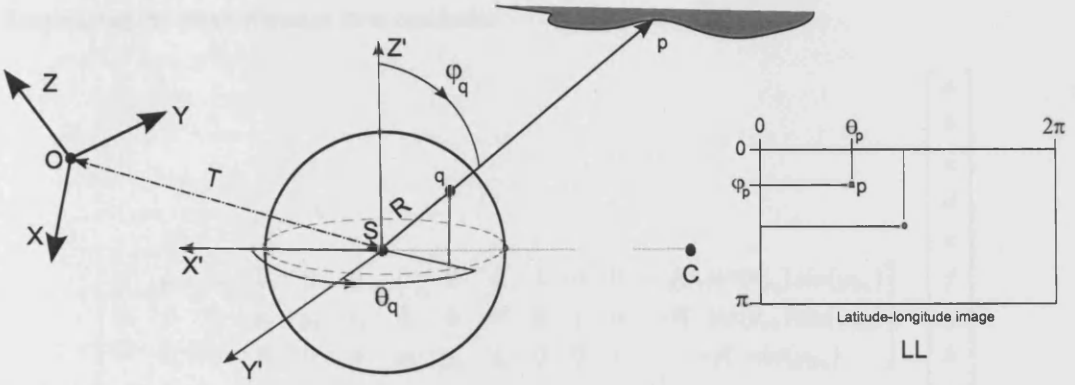


Figure 6.15: *Notations used to find the position of the lightprobe in the 3D geometric model. The position is defined by the transformation between the local coordinate system $X'Y'Z'$ and the world coordinate system XYZ.*

and $X'Y'Z'$ results in 3 equations. Therefore, knowing four corresponding points would solve the problem represented in equation 6.8. It should be noted that the problem can be solved with 3 known points, if the non-linear relation between the coefficients would have been considered.

For a point p in the 3D scene, its world coordinates are given as $p(x, y, z)$, while its local coordinates are given as $p(x', y', z')$. For a set of points p_i with $i \in [1, N]$ the coordinates $p_i(x, y, z)$ are available as the scene geometry is known. The local coordinates $p_i(x', y', z')$, however, are not available. On the other hand, the direction of $p_i(x', y', z')$ in spherical coordinates is available, because the projection q_i of the scene point p_i can be looked up in the latitude-longitude image LL. For each point p_i the spherical coordinates in the local coordinate system are given as:

$$\begin{bmatrix} x'_i \\ y'_i \\ z'_i \end{bmatrix} = s_i \cdot \begin{bmatrix} R\cos(\theta_{p_i})\sin(\varphi_{p_i}) \\ R\sin(\theta_{p_i})\sin(\varphi_{p_i}) \\ R\cos(\varphi_{p_i}) \end{bmatrix} \tag{6.9}$$

The scaling s_i is the same for the x, y and z axis, as the point p lies along the line Sq, where q lies on the sphere (radius R).

In other words, for each scene point p_i the world coordinates are defined and the local coordinates are defined up to a scaling factor s_i. This means that in combination with equation 6.8, each set of such corresponding points defines 3 equations, but creates one extra unknown. The system of 12 unknowns can therefore be solved if six corresponding points are known as this yields 12, defined by matrix W, plus 6, the scaling factors s_i, unknowns which equals to 18 unknowns and $6 \times 3 = 18$ equations. Equations 6.8 and 6.9 give the following set of 3 equations per point $p_i = (x_i, y_i, z_i)$:

$$s_i \cdot R \cdot \cos(\theta_{p_i})\sin(\varphi_{p_i}) = ax_i + by_i + cz_i + T_x \tag{6.10}$$

$$s_i \cdot R \cdot \sin(\theta_{p_i})\sin(\varphi_{p_i}) = dx_i + ey_i + fz_i + T_y \tag{6.11}$$

$$s_i \cdot R \cdot \cos(\varphi_{p_i}) = gx_i + hy_i + iz_i + T_z \tag{6.12}$$

Rearranging the terms in matrix form results in:

$$\begin{bmatrix} x_i & y_i & z_i & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -R \cdot \cos(\theta_{p_i}) \sin(\varphi_{p_i}) \\ 0 & 0 & 0 & x_i & y_i & z_i & 0 & 0 & 0 & 0 & 1 & 0 & -R \cdot \sin(\theta_{p_i}) \sin(\varphi_{p_i}) \\ 0 & 0 & 0 & 0 & 0 & 0 & x_i & y_i & z_i & 0 & 0 & 1 & -R \cdot \cos(\varphi_{p_i}) \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \\ T_x \\ T_y \\ T_z \\ s_i \end{bmatrix}$$

Setting up these equations for the 6 points results in:

$$Q \cdot V = 0 \tag{6.13}$$

with Q a 18×18 matrix:

$$\begin{bmatrix} x_1 & y_1 & z_1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -R \cdot \cos(\theta_{p_1}) \sin(\varphi_{p_1}) & \dots & 0 \\ 0 & 0 & 0 & x_1 & y_1 & z_1 & 0 & 0 & 0 & 0 & 1 & 0 & -R \cdot \sin(\theta_{p_1}) \sin(\varphi_{p_1}) & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 0 & 0 & 1 & -R \cdot \cos(\varphi_{p_1}) & \dots & 0 \\ \vdots & & & & & & & & & & & & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & x_6 & y_6 & z_6 & 0 & 0 & 1 & 0 & \dots & -R \cdot \cos(\varphi_{p_6}) \end{bmatrix}$$

and V a 18×1 matrix:

$$V^T = \begin{bmatrix} a & b & c & d & e & f & g & h & i & T_x & T_y & T_z & s_1 & s_2 & s_3 & s_4 & s_5 & s_6 \end{bmatrix}$$

Such a set of 18×18 equations can be solved using Singular Value Decomposition. The precision of the calculated sphere position can be enhanced by adding more corresponding points and solving the transformation matrix as the Least Squares Error (LSE) of the resulting set of equations.

As a summary, to find the position and orientation of the sphere within the frame of reference of the 3D scene the following steps need to be taken:

- Define 6 pairs (or more) of points $[p, q]$, with p a point in world coordinates, derived from the scene geometry, and q in local coordinates, derived from the pixel coordinate of p in the latitude-longitude image LL.
- These 6 corresponding points define 18 equations with 18 unknowns.
- Solve this system of 18 equations with 18 unknowns. The last row of the transformation matrix defines the position of the reflective sphere in the world coordinate system.

This is a relatively fast calibration procedure, requiring little manual input (the selection of corresponding points). Though it might seem incorrect to construct the point p_i from a latitude-longitude image generated by assuming that the scene lies at infinity, tests showed that when the corresponding points are appropriately chosen, the sphere's position can be retrieved accurately. This is further discussed in section 6.8.2. Once the position of the sphere is retrieved using the presented method, the accuracy of the position can be updated by taking into account the finite set-up of the scene. This second pass results in a more accurate position estimation. Nevertheless it has not been implemented in this chapter because it increases the computation time while providing little improvement.

C Absolute position of the camera in the world coordinate system

After solving the set of 18 (or more) equations, the transformation matrix W can be reconstructed. It was already discussed that the position of the camera to the sphere in the local frame of reference is given as $[-D, 0, 0]$, from this we can calculate the position of the camera $[x_C, y_C, z_C]$ as:

$$\begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix} = W^{-1} \cdot \begin{bmatrix} -D \\ 0 \\ 0 \end{bmatrix} \tag{6.14}$$

If required, the orientations of the X', Y' and Z' relative to XYZ axis, can be calculated in a similar way. In fact, the reconstruction of the local coordinate system can be simplified since the methods that describe the transformation from lightprobe image to latitude-longitude image, as outlined in this chapter, implicitly assume that the axis X' is equal to the viewing direction, and that the Y' and Z' axis are parallel to the image plane.

6.5.2 Sphere reflectance calibration

The reflective sphere used is often made of polished chrome-metal and will not reflect 100% of the scene radiance. This loss is in general approximated by a scaling factor, which can be calculated by comparing a reference object in the lightprobe image with its reflection in the sphere. This is illustrated in figure 6.16.

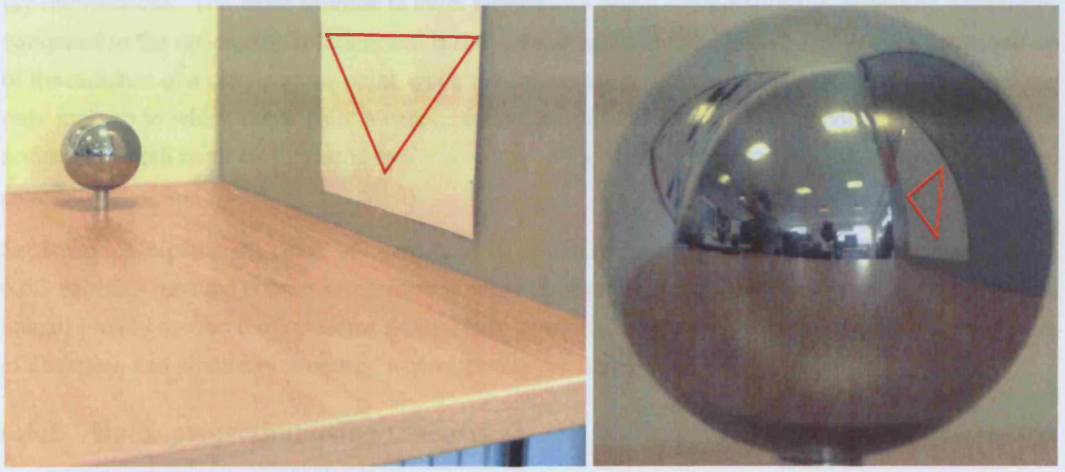


Figure 6.16: *Calibrating the reflectance of the lightprobe image: the relation between the white sheet of paper and its reflection in the lightprobe image gives the scaling factor in the RGB-colour channels.*

When ignoring fresnel reflection, the radiance reflected by the sphere should be identical to the radiance when captured by the camera directly. Debevec et al. [Deb04] also compensate for the fresnel refraction.

Based on the analysis and results presented in [Deb04] we have decided not to compensate for the fresnel refraction as the effect is only minor. The difference in reflection is larger near the outer ring of the lightprobe image, and it has already been argued that it is better not to capture important radiance in that section of the lightprobe anyway.

The scaling factor used to compensate for the non-specular behaviour of the sphere, does not only remove the imperfect specular reflection. It takes also care of the scaling that exists due to the optical vignetting. In chapter 2, equation 2.33 illustrates that the vignetting effect depends on the focal length and aperture width used to capture the HDRI. If the input image and the lightprobe image are captured with a different focal length or aperture width, this introduces an extra scaling on the pixel values.

6.6 Radiance registration

In a previous section it was mentioned that in its simplest form, inverse illumination could be approximated by a division: the diffuse reflectance of a point is given by the ratio of its radiance by the light that falls onto the point. To find this incoming light, the radiance of the lightprobe image needs to be re-mapped onto the scene geometry. The lightprobe image is positioned near the points for which it will be used to extract the incoming light, so that the radiance in the lightprobe image already gives an estimate of the incoming light. Nevertheless it is still important to *back-project* the radiance shown in a lightprobe image onto a scene to take appropriate care of the surface orientations and occlusion effects that might take place.

Since the scene geometry and the positions of the camera and lightprobe are known, it is possible to warp the radiance values onto the scene geometry in a mathematically accurate manner. There are two ways to achieve this. The first shoots rays through a pixel of the lightprobe image, calculates its point of reflection and finds the intersection of the reflected ray with the triangles of the 3D scene. This is a computationally expensive solution where the computation time depends on the complexity of the 3D scene. The second calculates for each scene point p its reflection point on the sphere by solving a 4th-degree self-inversive polynomial and using simple OpenGL commands. From this reflection point its projection in the lightprobe image is found without the need to perform computationally expensive ray intersections. The latter solution is more elegant and offers computationally appealing advantages compared to the ray-tracing solution, and is therefore adopted in this chapter. It allows an easy look-up of the radiance of a certain scene point, while the former method operates the other way around and can only look up to which scene point a certain radiance value belongs to. It is important to note that the accuracy of both methods is limited by the accuracy of the scene reconstruction and lightprobe position calibration.

Section 6.6.1 explains the usage of OpenGL as an identifier during the back-projection process, section 6.6.2 explains how the correct scene point is mapped onto the correct radiance value in the lightprobe image. Finally section 6.6.3 presents a case-study in which a numerical estimate is given of the amount of distortion and pinching introduced when ignoring the finite set-up of a 3D scene.

6.6.1 Back-projection using OpenGL

Figure 6.17 gives an overview of the five different steps that the back-projection comprises. This section covers each of these steps. The goal of these steps is to create for each lightprobe m *back-projection textures* BT_{ijm} for the illumination clusters IC_{ij} and neutral cluster N_{ij}. These back-projection textures, when mapped onto the geometric model of the scene, recompose the scene radiance distribution at the capture time of lightprobe m. It is assumed that for each lightprobe image LP_m a latitude-longitude

image LL_m is available. In this section we make abstraction of how this latitude-longitude image is generated, the actual generation is explained in more detail in section 6.6.2.

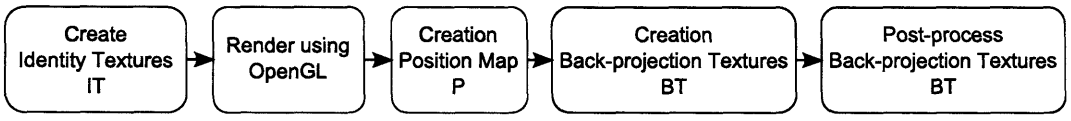


Figure 6.17: *Overview of the back-projection procedure that consists of five different steps.*

To back-project the radiance values from a lightprobe m onto the scene points in the scene, we need to know to which scene point a pixel in LL_m is projected. This is achieved through generating a *position map* P_m for a lightprobe m that is aligned with the image LL_m. The position map P_m has the same dimensions as LL_m and consists of 4 colour channels $RGB\alpha$ denoted as $[P_m(k,l,0), P_m(k,l,1), P_m(k,l,2), P_m(k,l,3)]$. The quartet defined by $P_m(k,l)$ uniquely defines the scene point p onto which the radiance in $LL_m(k,l)$ should be projected. The first two channels $P_m(k,l,0)$ and $P_m(k,l,1)$ define the scene triangle to which the point p belongs. The last two channels $P_m(k,l,2)$ and $P_m(k,l,3)$ define the the position of p inside the triangle.

Such a position map is generated by rendering the scene using a colour coded texture onto a cube which can be transformed to a longitude-attitude map with the same dimensions as LL_m. The colour coded texture shares the same 4 colour channel representation as the position map. The colour coded texture is created through a set of *identity textures* IT. Each illumination and neutral cluster receives such an identity texture. It has the same dimension as the original texture assigned to the illumination and neutral clusters. Before generating the identity texture, each scene triangle receives a unique identity number. The identity texture creation proceeds then as follows. For an illumination cluster IC_{ij} each pixel in its associated identity texture IT_{ij} receives the following colour quartet $[S_x, S_y, T_x, T_y]$:

- $[S_x, S_y]$: 32-bit R- and G-channel: These two floating point channels represent the identity number of the triangle to which the pixel belongs to. This means that $2^{64} = 1.8 \cdot 10^{19}$ different triangles are allowed in the scene, usually quite an acceptable number.
- $[T_x, T_y]$: 32-bit B- and α-channel: These two channels identify the position of the pixel in the texture. To obtain this, the B-channel is equal to the x-coordinate of the pixel in the texture, the α-channel is equal to the y-coordinate of the pixel in the texture.

The identity textures for the neutral clusters are generated in a similar manner. After creating IT_{ij}'s for all clusters in the scene, the scene is rendered with the OpenGL texturing functions enabled. The scene is rendered onto 6 planes defining a cube around the sphere i.

The resulting cube map is further transformed to a latitude-longitude map resulting in the position map P_m. The first two colour channels of a pixel in P_m define the triangle to which the pixel belongs to, hence the illumination cluster. The other two colour channels define the position of the pixel inside the triangle, as these can be derived from the texture coordinates of the point using the barycentric coordinates. This can be seen as follows. When using texture mapping to map a texture onto the scene geometry, a triangle receives its radiance values through a set of three texture coordinates. Each point p inside a triangle defined by three points p_1, p_2, and p_3 can be represented by three normalised barycentric coordinates $[w_1, w_2, w_3]$ that define its position in the triangle as follows: $p = w_1p_1 + w_2p_2 + w_3p_3$. The same barycentric coordinates can be used to find the texture coordinates t of point p inside the texture, using the texture coordinates t_1, t_2, and t_3 of respectively p_1, p_2, and p_3 as follows: $t = w_1t_1 + w_2t_2 + w_3t_3$.

The opposite holds as well: from the texture coordinates t of a point inside a triangle, its 3D position p can be derived using the barycentric coordinates that define that texture coordinates t based on the texture coordinates t_1, t_2, and t_3 of the points that define the triangle in the texture.

The creation of the identity texture IT_{ij} is graphically illustrated in figure 6.18 (a). For reasons of simplicity, the illumination cluster IC_{ij} consists of one triangle with identity number X', the radiance texture of the illumination cluster contains the radiance of this triangle, among maybe other scene triangles as is shown in figure 6.18 (a). The $RGB\alpha$-channels of a pixel are derived from the identity number X' of the triangle and the texture coordinates, as is illustrated. Figure 6.18 (b) illustrates how the projection map is created through rendering the identity textures onto a cube centred around the sphere. The cube map is in fact generated by rendering the scene six times, once on each face of the cube. The 2D interpretation of the scene consists of one neutral cluster, two material clusters and three illumination clusters.

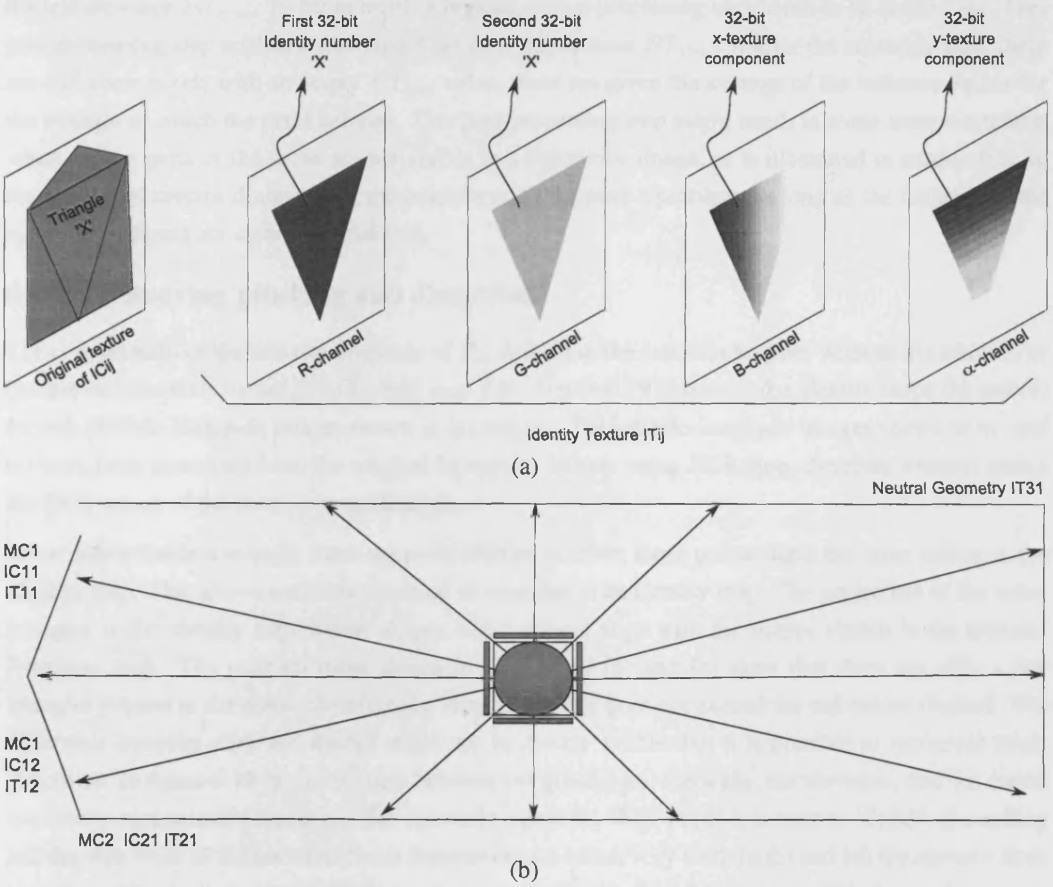


Figure 6.18: *(a) Creation of the Identity Texture IT_{ij}. In this example an illumination cluster consists of one triangle. The identity texture consists of four channels, which are derived from the triangle's identity number X, and the position of a pixel in the texture of its illumination cluster. The R-channel contains the first 32-bit of the unique identity number X of the triangle, while the G-channel contains the last 32-bit of the unique identity number of the triangle. The B-channel contains the x-component of a pixel's position in the texture of the illumination cluster. The α-channel contains the y-component of a pixel in that texture. (b) Creation of the position map: the scene is rendered onto the six faces (four are shown here in this 2D interpretation) of a cube centred around the sphere.*

The transformation from cube map to latitude-longitude map proceeds in the following manner. While looping through the pixels in the position map the spherical coordinates of the pixel on the sphere are reconstructed from the coordinates of the pixel in the position map. The direction created as such, is

used to find the projection of the pixel on the cube. The cube map is spatially discretized too, and forms a grid of sampling points. To retrieve the values rendered onto the cube, bi-linear interpolation is used. A transformation from cube map to latitude-longitude image is not uniform. To remove aliasing, the cube map should be pre-filtered with a low-pass filter.

To perform the the actual back-projection, another floating point texture is created per illumination and neutral cluster called *back-projection texture* BT_{ijm}, with i again an index into the material clusters, j an index into the illumination and neutral clusters, and m an index defining the lightprobe image under consideration. The back-projection of lightprobe m with position map P_m generated using the process defined previously, proceeds by looping through the pixels t in the position map P_m. The colour of pixel t in P_m defines a position s in the original radiance texture of the illumination cluster to which t belongs. The radiance of s retrieved from LL_m is now stored at that position s in BT_{ijm}.

The creation of the back-projected textures BT_{ijm} as outlined above does not fill in all texture values in the texture maps BT_{ijm}. To fill in missing regions, a post-processing step needs to be carried out. This post-processing step applies a smearing filter over the textures BT_{ijm}. If after the smearing step, there are still some pixels with an empty BT_{ijm} value, these are given the average of the radiance values for the triangle to which the pixel belongs. This post-processing step might result in some texture artefacts when certain parts of the scene are not visible in a lightprobe image, as is illustrated in section 6.8. In the context of inverse illumination, these artefacts do not pose a problem, as long as the radiance of the main light sources are clearly represented.

6.6.2 Removing pinching and distortion

The combination of the first two channels of P_m defined in the previous section, without the addition of the last two channels, is called an *identity map* IM. Figure 6.19 illustrates the identity maps (b) and (d) for two latitude-longitude images shown in (a) and (c). The latitude-longitude images shown in (a) and (c) have been generated from the original lightprobe images using HDRshop, therefore without taking the finite set-up of the scene in consideration.

Since points inside a triangle share the same identity number, these points share the same colour in the identity map. This allows easy identification of triangles in an identity map. The projection of the scene triangles in the identity map defines shapes which should align with the shapes visible in the latitude-longitude map. The position maps shown in figure 6.19 (b) and (d) show that there are only a few triangles present in the scene, therefore the identity number does not exceed the red colour channel. The difference between some red shades might not be clearly visible, but it is possible to recognize some structures. In figure 6.19 (b) the borders between the ground and the walls, the elevators, and the doors, are clearly recognizable (these are also shown in figure 6.1 (b)). In (d) it is easy to identify the ceiling and the side walls of the elevator. Some features do not match very well: in (c) and (d) the elevator floor (metallic with black circles) does not overlap very well with the identity map. This is nearly entirely due to the pinching and distortion effects, though small misalignments in camera and sphere calibration may also contribute to the error. This allows us to make the following conclusion. The latitude-longitude images, generated while ignoring the finite set-up, distort the scene radiance projected onto the lightprobe images. When used in combination with a position map to back-project the scene radiance contained in a lightprobe image onto the geometric model of the scene, as outlined in section 6.6.1, this results in a distorted back-projection.

To remove the distortion and pinching effects, a novel latitude-longitude transformation is developed that uses the 3D scene geometry, the sphere and camera position and the lightprobe image to create a latitude-longitude image LL_m for lightprobe m. This distortion-free transformation can be incorporated as an

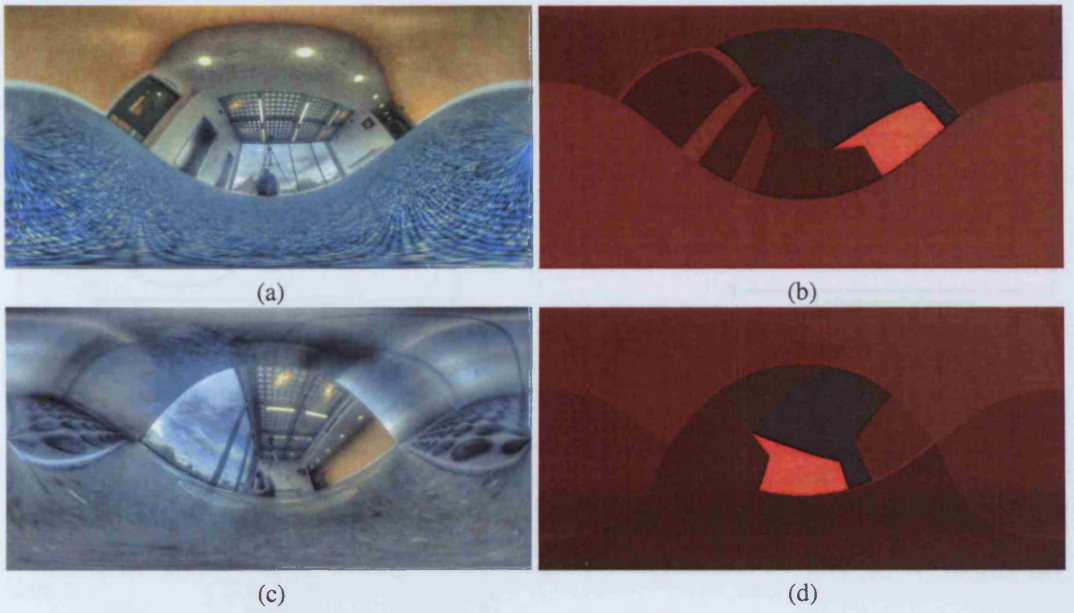


Figure 6.19: *Using OpenGL to map the radiance onto the geometry: the left column (a,c) shows two latitude-longitude images belonging to different lightprobe images captured from two different positions. The latitude-longitude images are generated using HDRshop and therefore show distortion and pinching. The right column (b,d) shows the corresponding identity maps. Pixels with the same colour belong to the same illumination cluster. The pinching is clearly visible on the sides of the latitude-longitude images shown in (a) and (c).*

extra module in the framework presented in figure 6.17. In fact it needs to be executed after the creation of the position map P_m, but before the creation of the back-projected textures. The new back-projection pipeline is displayed in figure 6.20.

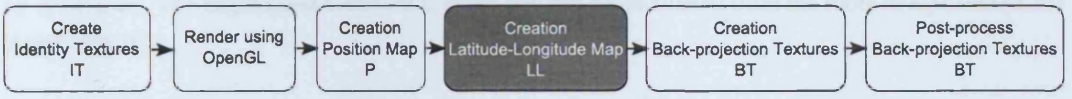


Figure 6.20: *Overview of the back-projection procedure including the distortion removal.*

Finding the corresponding pixels in the lightprobe image for the 3D scene points is equivalent to solving *Alhasen's Billiard problem*[Elk65][Neu98][Wal92], for which a 2D graphical interpretation is given in figure 6.21. The following terminology applies: δ is the angle between the rays from the centre of the sphere towards point p and the camera C, ψ is the angle of reflection, and α_t is the angle of the reflected ray towards the centre of the camera. The Alhasen Billiard problem is equal to finding the angle ψ at which a ray needs to be *shot* towards a specular circle in order for it to be reflected to a certain point (the camera). It resembles similarities to the problems associated with the billiard game, albeit if the billiard table was circular.

It was recently proven that ψ cannot be solved from δ, R, p and C (or D) using the Ruler and Compass method [Neu98]. Several different methods have been developed that succeed at finding the angle ψ, some using extensive iterations. The most elegant available in the literature and the easiest to implement in the context of the application presented in this chapter is that described in [MV91], which calculates $\epsilon = \frac{\pi}{2} - \psi$ from the roots of the following self-inversive polynomial:

$$\alpha z^4 + \beta z^3 + \gamma z^2 + \overline{\beta} z + \overline{\alpha} = 0 \tag{6.15}$$

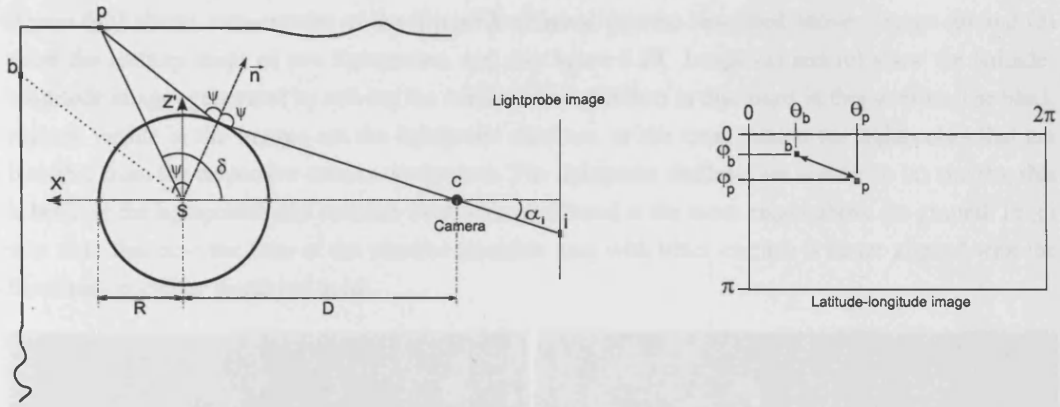


Figure 6.21: *Left: Generating a latitude-longitude image by deriving the angle ψ from p, S, C and D. Right: the distortion that occurs when the latitude-longitude image is derived by assuming the scene to lie at infinity.*

with

$$\alpha = e^{i\delta}(e^{i\delta} - k_1 k_2) \tag{6.16}$$

$$\beta = k_1^2 + k_2^2 - 2k_1 k_2 e^{i\delta} \tag{6.17}$$

$$\gamma = 2(k_1^2 + k_2^2 - k_1 k_2 \cos(\delta) - 1) \tag{6.18}$$

$$k_1 = \frac{R}{D} \tag{6.19}$$

$$k_2 = \frac{R}{\sqrt{p_{z'}^2 + p_{x'}^2}} \tag{6.20}$$

The scene point p is represented in the local coordinate system $X'Y'Z'$, where p lies in the plane formed by X' and Z'. The angle ϵ and therefore ψ exists and can be calculated from the roots of equation (6.15) if and only if:

$$0 < \delta \leq cos^{-1}(k_1) + cos^{-1}(k_2) < \pi \tag{6.21}$$

We can calculate ϵ now as:

$$\epsilon = \frac{1}{2}\cos^{-1}(Re(z)) \tag{6.22}$$

This results in four possible values for ϵ, but only ϵ for which the following condition applies is the correct reflective angle:

$$\delta + 2\epsilon = \cos^{-1}(k_1 \cos(\epsilon)) + \cos^{-1}(k_2 \cos(\epsilon)) \tag{6.23}$$

Once ψ is known, α_t is calculated using the sine rule:

$$\frac{D}{\sin(\pi - \psi)} = \frac{R}{\sin(\alpha)} \tag{6.24}$$

This can easily be extended to 3D by executing the above calculations for each scene point p defined in the position map P_m in the plane formed by the camera, the sphere centre and p. Now for each sphere positioned in the scene, and from each lightprobe image LP_m a correct radiance map is created in latitude-longitude format, also called LL_m. This LL_m now represents a good estimate of the radiance of a point p as seen from the centre of the sphere.

Figure 6.22 shows some results of the distortion removal process described above. Image (b) and (d) show the identity maps of two lightprobes, see also figure 6.19. Image (a) and (c) show the latitude-longitude images generated by solving the mathematical problem as discussed in this section. The black regions visible in the images are the lightprobe *shadows*, or the areas behind the lightprobes that are invisible from the respective camera viewpoints. The lightprobe shadows are similar in (a) and (b), this is because the lightprobes and cameras used were positioned at the same height above the ground. In (c) it is clear that now the floor of the elevator (metallic grey with black circles) is better aligned with the illumination cluster displayed in (d).

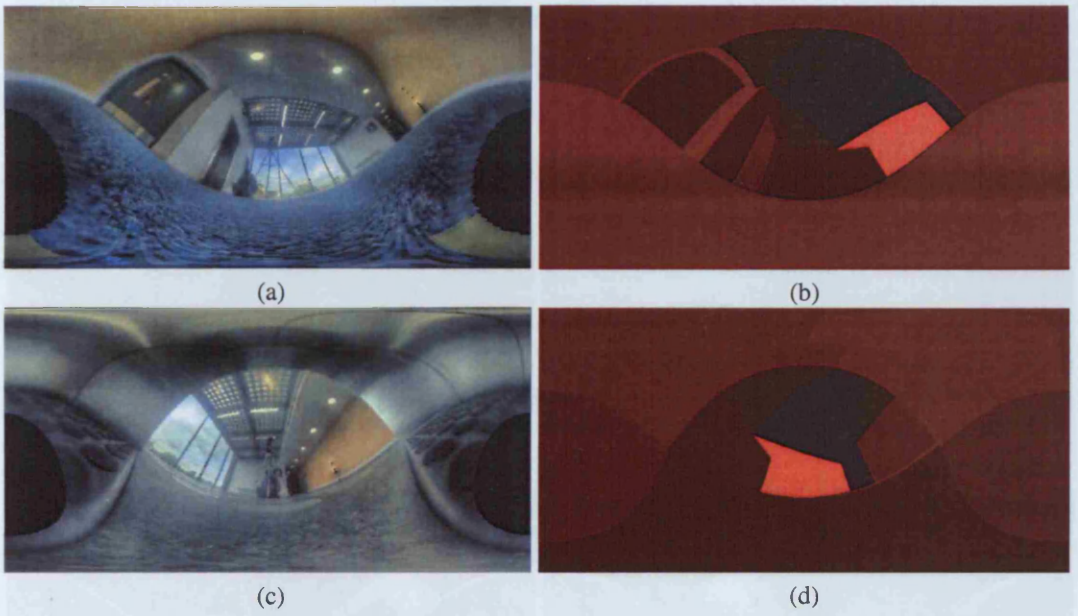


Figure 6.22: *Removing distortion by solving Alhasen's Billiard Problem. (b) and (d) show the same identity maps as in figure 6.19. (a) and (c) are generated using the method described in section 6.6.2. The large black areas in (a) and (c) show the* shadow *created by the lightprobes: the volume behind the sphere is not visible from the viewpoint of the camera. The identity map shows that some features are better aligned after the distortion removal: the floor in the elevator (grey with black circles) is better aligned with the floor shown in the identity map.*

6.6.3 Numerical analysis of distortion and pinching

The amount of distortion and pinching introduced by ignoring the finite set-up of the scene depends on the dimensions and the geometry of the 3D scene. In this section a numerical analysis is given of the distortion for the scene shown in figure 6.1, which consists of a hallway and two elevators. Various lightprobe images were captured inside this scene, some were already presented in figures 6.19 and 6.22.

The amount of distortion is calculated as the distance between scene points p and b, where b is the scene point to which the radiance of scene point p would be projected if the finite set-up of the 3D scene is ignored, as was explained in figure 6.8 and is also illustrated in figure 6.21. This distortion distance is calculated for all scene points and is stored in a distortion map calculated in the following manner:

1. Calculate for scene point p the position of the pixel t at which the radiance of p is reflected by the reflective sphere.
2. For pixel t find its corresponding scene point b, at which it would have been reflected if the scene was assumed to be infinitely far from the centre of the sphere. This is implemented using a ray-tracer.

3. Calculate the distance of $|pb|$ and store this value at (ϕ_p, θ_p) in a latitude-longitude image.

4. If p falls inside the lightprobe's shadow, b cannot be calculated, and the corresponding distance is set to 0.

Figure 6.23 shows two latitude-longitude images (a) and (b) and the corresponding distortion images (c) and (d), generated using the steps outlined above. The maximum amount of distortion is $5.11m$ in (c) and $4.54m$ in (d). The scene dimensions are $\approx 3m \times 6m \times 12m$. As expected, the distortion is the highest on planes that are parallel to the viewing directions. Also, the distortion is quite large in areas containing neighbouring or connecting surfaces with different orientations (e.g, elevator cubicle), and at the boundaries of occluding objects. Some quantization errors are visible (circular patterns), and are due to the quantization effects in the ray tracer.

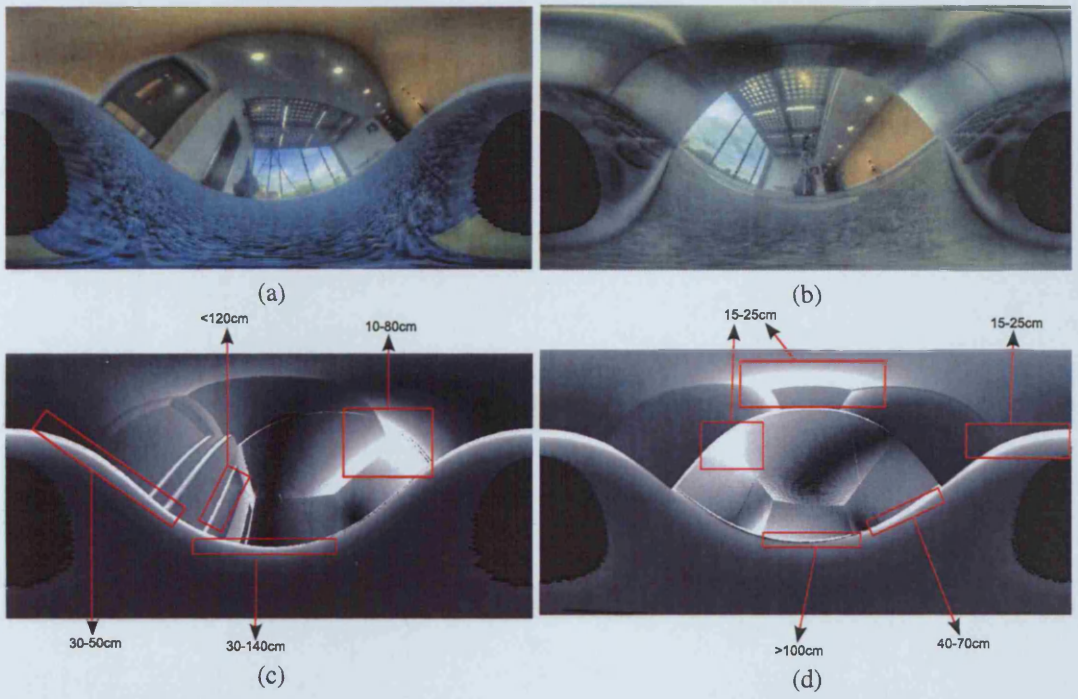


Figure 6.23: *Analysing the error introduced by distortion. Two latitude-longitude images (a) and (b) and their distortion images (c) and (d), generated using the method outlined in section 6.6.3. The brightness of the scene is an indicator of amount of distortion. The maximum distortion is* $5.11m$ *in (c) and* $4.54m$ *in (d). The scene dimensions are* $\approx 3m \times 6m \times 12m$.

The distortion images indicate several significant issues. When important scene features, such as light sources, are positioned on a surface parallel to the viewing direction, or near boundaries of occluding surfaces, significant distortion can be introduced. Therefore, when capturing the lightprobe images, care needs to be taken to position the camera such that significant distortion is reduced for important scene features.

6.7 Inverse Illumination

As the main contribution of this chapter lies in the radiance capture methodology, the reflectance calculation was limited to diffuse-only materials. Nevertheless a more sophisticated BRDF model, with anisotropic and specular characteristics, could also be considered. The BRDF calculation presented here follows a similar iterative strategy as presented by Boivin et al. [BG01]. The methodology has been

discussed in chapter 2, another more detailed overview is presented in figure 6.24 which shows that the entire inverse illumination process can be split into two parts.

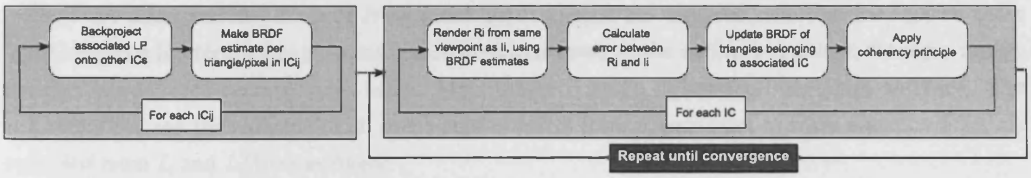


Figure 6.24: *An overview of the inverse illumination process.*

The first part makes an initial estimate of the BRDF for each triangle or pixel inside an illumination cluster. This estimate is calculated from the back-projected textures of its associated lightprobe LP_i, the scene radiance available in the input images I_i, and the geometry. Then an iterative process updates the previous BRDF estimate. The update is steered by the difference between I_i and R_i, where the latter is an image rendered from the same viewpoint as I_i, using global illumination and the current BRDF estimates. The illumination during this rendering phase is simulated with local textured area light source, where the textures are the back-projected textures of the neutral cluster, generated using the method described in section 6.6.1. After the update step, the BRDF values within a material cluster are further updated using *the coherency principle*. This process continues until convergence.

Section 6.7.1 discusses how the BRDF is calculated at triangle level, while section 6.7.2 presents a possible per-pixel BRDF estimate. In section 6.7.3 the update step using an error measure is explained. Section 6.7.4 shows how the BRDF estimation is distributed to the different MC_i using *coherency*. Finally, section 6.7.5 discusses the renderer used to render the images R_i.

6.7.1 Reflectance estimation per triangle

Inverse illumination finds a BRDF estimate for all materials in a scene. Usually a scene consists of triangles, and different materials can appear on one modelled triangle. Therefore it seems a logical choice to make a BRDF estimate for each scene point, as this would preserve structure and texture of the scene. Practically this is less attractive, because the computation time would be too high for large scenes. Therefore, the scene is often triangulated, for instance based on the variance of the radiance in a triangle, to retrieve a minimal triangle grid. Then, to speed up the calculations, a BRDF estimate is made per triangle. It is this approach that was adopted and is explained in this section.

Section 2.4 already provided a framework of inverse illumination, and the method presented here follows this framework. The BRDF is calculated iteratively, first a start BRDF value ρ^0 is calculated from the radiance equation which describes the reflected radiance for a point p into direction ω_r as:

$$L_r(p,\omega_r) = \int_\Omega \rho(\omega_i,\omega_r) L_i(p,\omega_i)\cos(\theta_i)d\omega_i \tag{6.25}$$

where L_i is the incident radiance at p in direction ω_i and ρ is the general reflectance, and the emittance $L_e(p)$ is ignored, as it is most likely zero. This simplifies to an explicit equation for diffuse material to the reflectance ρ_d:

$$\rho_d = \rho^0 = \frac{L_r(p,\omega_r)}{\int_\Omega L_i(p,\omega_i)\cos(\theta_i)d\omega_i} \tag{6.26}$$

In other words, a good estimate for the diffuse BRDF for a point p is given as a ratio where the nominator is the radiance value of the point p and the denominator is the sum of the radiance values, modulated

by a cosine, that can be seen from point p. Please note that these equations are actually split into three components for the three RGB-channels, as discussed in chapter 2.

Section 6.4 discussed that a set of N different input images are captured, and that for each of these input images a lightprobe image is associated positioned near to the surfaces visible in the input image. In other words, each scene point p has an input image I_i and a lightprobe image LP_i assigned. The radiance $L(p)$ and the radiance of all scene points visible from p, necessary to solve equation 6.26, are extracted from I_i and LP_i respectively.

In practice, the denominator is derived by rendering the scene on a hemisphere positioned around p, where the radiance values of the scene points come either from I_i (when visible in I_i) or LP_i (when not visible in I_i), and by summing all resulting pixels after weighting them with the appropriate functions. As the BRDF is only calculated per triangle, the starting BRDF value ρ^0 is calculated for each midpoint (centre) of a triangle using equation 6.26.

When a scene is not perfectly diffuse, the above described estimate will deviate from its actual value. Other effects like geometric inaccuracies also contribute to the deviation of the estimate of the BRDF from its actual value. Therefore the BRDF is further updated, by calculating a measure E of the error between the actual value and the estimate. The BRDF is refined as follows:

$$\rho_d^1 = E \times \rho_d^0 \tag{6.27}$$

where ρ_d^0 is the initial estimate and ρ_d^1 the updated BRDF value. When this process is repeated iteratively, the update step can be re-written as:

$$\rho_d^{n+1} = E \times \rho_d^n \tag{6.28}$$

ρ_d^{n+1} is the BRDF at iteration $n+1$ and ρ_d^n at iteration n.

The error measure E is calculated by comparing the radiance values of the triangle under consideration in its associated input image I_i with those in a rendered image R_i. This image R_i is rendered from the same viewpoint as I_i, with all materials set to their corresponding estimates and using the back-projected textures of LP_i on the neutral cluster as textured area light sources. Mathematically the error measure is given as:

$$E = \frac{I_i^{avg}}{R_i^{avg}}, \tag{6.29}$$

with I_i^{avg} and R_i^{avg} the average of the radiance values of the triangle under consideration in respectively I_i and R_i. When $I_i^{avg} \approx R_i^{avg}$, the error E will approximately be equal to 1 and $\rho_d^{n+1} \approx \rho_d^n$. In other words the BRDF values converge. When $I_i^{avg} \neq R_i^{avg}$, the above described error estimate will steer the update into the correct direction. Suppose that the current BRDF estimate is too low, this would result in a too dark rendering, or $\frac{I_i^{avg}}{R_i^{avg}} > 1$. Consequently, the update ρ_d^{n+1} will be larger than its previous value. A problem occurs when the BRDF is steered towards a value larger than 1. In that case, PBRT clips the BRDF to 1. As a consequence the iterations on the BRDF would never converge, as in each iteration the BRDF would be forced to be larger than its previous value (E is always larger than 1). We have solved this problem by allowing a BRDF larger than 1 but below a certain threshold ($= 1.2$) after the error update, and clipping the BRDF to 1 before the rendering. After all, due to un-modelled specular effects, the BRDF could artificially be larger than 1 to compensate for the lack of specular highlights. Another reason is that the scaling factor applied to the reflective sphere, as defined in section 6.5.2, might be too low, which is compensated by larger BRDF values. If the BRDF becomes larger than 1.2, the iterations are stopped and it is assumed that the scaling factor applied on the lightprobe images is unsuitable. More details about how the error estimates is made, is given in section 6.7.3.

6.7.2 Per pixel reflectance estimation

Calculating a BRDF estimate per triangle removes some texture features which are patterns visible in objects such as wood, carpet, posters and book covers. If it is necessary to preserve these features it is preferred to calculate a BRDF per pixel visible in the texture associated with the triangles in the scene. The per-pixel BRDF estimations are carried out on the material clusters labelled as being patterned, as defined in section 6.4.

For material clusters labelled as being patterned, the BRDF is calculated at pixel level[2]. A starting BRDF value is calculated in a similar approach as in section 6.7.1. The irradiance for each triangle is estimated through rendering the scene radiance on a hemisphere positioned at the centre of the triangle, and summing these values after modulation by a cosine. For each pixel in the triangle, the BRDF is estimated by dividing the radiance value by the gathered irradiance.

Instead of storing the BRDF values per triangle, they are now stored in a *diffuse reflectance map (dm)*. It has the same dimensions as the radiance texture of the illumination clusters. Each triangle has texture coordinates that refer to the original radiance texture, the same texture coordinates are now used to retrieve the BRDF of a point inside the triangle from dm. For non-patterned triangles all pixels in dm have the same BRDF value. For patterned triangles, the entries in dm are the per-pixel calculated BRDF values. This dm can effectively be used by PBRT to render the scene with the defined diffuse BRDF.

The update step used to refine the BRDF estimate is similar to that used in section 6.7.1, except for that now each BRDF estimate for each pixel is updated using the previously defined error measure. The error measure is still calculated as an average over the triangle. The reason must be found in the specific renderer used in this chapter, see section 6.7.5. The renderer PBRT[PH] is a stochastic renderer, and a trade-off needs to be made between rendering quality and rendering time. If the rendering time is reduced, the rendered image is noisy which can result in noisy update steps when calculated per pixel. In this chapter, the trade-off resulted in rendering lower quality renderings for the iterations and calculating the error-update per triangle.

6.7.3 Updating the BRDF using a measure for error

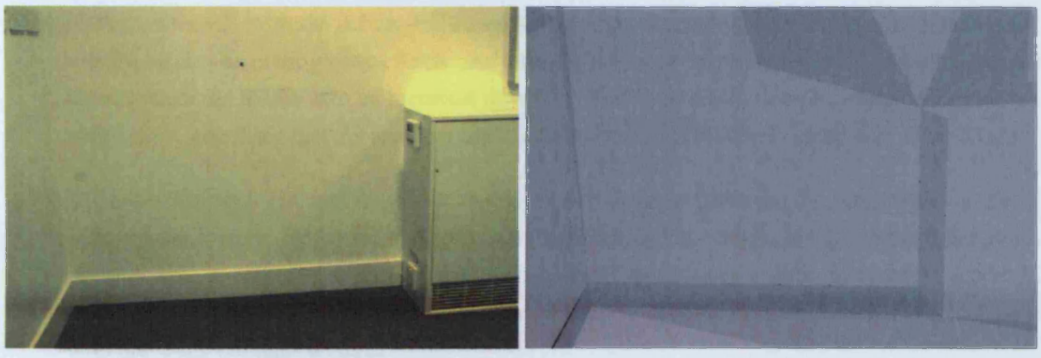
The error described in section 6.7.1 is calculated per triangle. It is given as the ratio of the average of radiance values for the pixels covered by the triangle in the input image I_i by that for the rendered image R_i. The rendered image is obtained by rendering the scene from the same viewpoint as I_i, using the current BRDF estimates for the scene triangles. The light sources are defined by textured area light sources, where the textures are equal to the back-projected textures of LP_i onto the neutral cluster. Rendering a scene as described here results in the same final rendered image as when all materials (including the surfaces of the neutral cluster) in the scene have a BRDF estimate and only the actual light sources are activated, because both configurations should converge to the same stable solution defined by the radiance equation.

The area that the triangle covers in the input image (and therefore in the rendered image) is found by rendering an identity image, similarly to that defined in section 6.6 used to calculate the back-projected textures. The identity image is created by rendering the scene from the same viewpoint as I_i; the triangles are rendered in an RGB colour triplet $[T_i, T_j, 0]$, where the first two channels define the unique identity number of the triangle. The blue channel is zero, since in section 6.6 the unique identity number was also only rendered using the R- and G-channel. The resulting identity image defines exactly which triangle

[2]In fact, labelling MCs as patterned might not be efficient as they sometimes extend a wide area, while a scene feature might cover only a small area. Labelling triangles could result in a better distribution of the patterned areas in a scene.

each pixel belongs to in I_i and R_i. The average radiance values necessary for the error estimate from equation 6.29 are calculated through looping over the images I_i and R_i.

An example of an identity image and its corresponding input image are given in figure 6.25, in which every triangle has a unique colour value.



(a) (b)

Figure 6.25: *An input image (a) and its identity image (b). Every triangle has a unique colour value (here tonemapped to greyscale values). The identity image can be used as a look-up table to find which triangle a certain pixel belongs to in R_i or I_i.*

6.7.4 Coherency to remove geometrical errors

With the method described above, all triangles receive a different BRDF. Ideally the BRDF estimate is equal among the triangles belonging to the same material cluster. Local changes however, such as variance in texture over a triangle, specular highlights, and geometrical errors, ensure that the BRDF estimate is not the same across the different triangles of a material cluster. Nevertheless, the organization of triangles into material clusters can be used during the BRDF estimation to create an homogeneous, or coherent, BRDF among the triangles belonging to the same material cluster.

In order to obtain this coherent BRDF, the BRDF values for all triangles belonging to the same *non-patterned* MC are averaged using a specific weighting function which reflects the accuracy to be expected for the calculation of the BRDF for a certain triangle. The weighting occurs after each rendering as is shown in the overview shown in figure 6.24.

In this chapter, this weighting function is given by the relative distance of a triangle to the lightprobe image used to calculate its BRDF. This makes sense as the radiance representation extracted from a lightprobe image is more accurate for points near the reflective sphere used to capture the lightprobe image. In mathematical terms, the BRDF of MC_i is given as:

$$\rho_i = \sum_{t=1}^{T} w_t \times \rho_t \tag{6.30}$$

$$w_t = \frac{d_t^{-1}}{\sum_{j=1}^{T} d_j^{-1}} \tag{6.31}$$

with T the total number of triangles belonging to MC_i and d_t the distance of triangle t to its associated reflective sphere. To improve the weighting function, the radiance variance of a triangle could be incorporated because the per triangle BRDF estimate is less accurate for triangles with a large texture variance. Another factor to be incorporate could be the geometrical accuracy expected, though the weighting in itself will already remove local errors due to geometric inaccuracies.

Substituting the BRDF for all triangles by the coherent BRDF calculated above, ensures a uniform BRDF across all triangles belonging to the same MC. This coherency principle cannot be applied to the material clusters with a patterned texture. During the experiments, as is discussed in section 6.8, a few practical issues were raised:

- Diffuse estimate: with the diffuse-only assumption, the rendered images R_i do not contain specular highlights or other glossy effects. As a result, the error for triangles lying in such specular zones, steers the BRDF into an incorrect direction. The coherency principle, effectively removes such errors, assuming that the specular highlights cover only a relatively small area in an image.

- Geometrical errors: not applying the coherency principle on the patterned clusters, creates artefacts between the triangles because the error-update is performed at triangle level. On the other hand, the per-pixel update step is in fact erroneous in itself when only a poor geometrical model is available. Similarly to the problems outlined in chapter 4 the error updates will be misaligned with the scene radiance texture.

Following this discussion, we can state that applying the coherency principle effectively removes errors from un-modelled specularities and an inaccurate geometric reconstruction. The coherency principle cannot be applied to patterned surfaces, which may result in artefacts.

6.7.5 Rendering using PBRT

The images R_i are rendered with PBRT using Monte Carlo path tracing [PH]. We have extended the standard version of PBRT to support textured area light sources and allow importance sampling of these. Since the illumination is based on the entire back-projected radiance on the neutral clusters, it is important to specifically adjust the path tracer to provide more samples where the radiance is high to avoid noise in the rendering. The original path tracer implemented in PBRT determines the contribution from direct illumination by randomly choosing a light source to sample and chooses sample points uniformly over the area of the light source. This approach results in high variance of the Monte Carlo estimator because the HDR textures used as area light sources contain both low and high radiance values (very high variance). The importance sampling is implemented by re-triangulating the area light sources, and calculating the contribution from each new "small" area light source (average emittance times area) to build a discrete 1D cumulative density function (CDF). This CDF is then used to determine which area light source to sample when estimating the contribution from direct illumination, resulting in a lower-variance Monte Carlo estimator.

The PBRT path tracer is slow, and the computation time increases with the number of samples per pixel used. To speed up the calculation, we varied the samples per-pixel from 1 to 2024. The iterations were usually completed with 32 to 256 samples per pixel. The final relit images were calculated using 2024 samples per pixel.

The updates to PBRT were implemented by two MSc. students for their Masters thesis [NV06].

6.8 Results for relighting applications

After a successful inverse illumination, the scene can be relit using a novel illumination pattern. It suffices to back-project a different lightprobe image onto the neutral geometry and then render the scene, similarly to how the rendered images R_i were generated in section 6.7.

In this section, two different relighting scenarios are analysed. The methodology presented in section 6.3 has been applied to each of these two scenarios. The first scenario, see section 6.8.1, applies the method on a synthetic scene. Such a scene allows to analyse the different modules of the relighting method separately. More precisely it allows to test the back-projection using ideal lightprobes, and calculate and verify the perfectly diffuse BRDF values of the scene objects. The second scenario operates on a real-world environment and tests the usability of the method on real image data, see section 6.8.2.

To aid the analysis of the presented method, a graphical user interface was built, called REFLECT. REFLECT combines easy scene navigation, parameter control, semi-manual lightprobe calibration, inverse illumination and relighting into one system[3]. Throughout the following sections, several screen shots of REFLECT are shown. For more details about the system, the reader is referred to appendix E.

6.8.1 Results for a synthetic scene

The methodology is applied to a synthetic scene, for which the geometry and lightprobe positions are accurately known. This allows verifying the correctness of the radiance back-projection algorithm. The scene is diffuse and the diffuse parameters of the scene are known. The difference between the BRDF estimates and the true BRDFs gives a measure for the accuracy of the inverse illumination calculations, without the errors that may be introduced due to the lightprobe calculations. The following sections discuss the set-up of the scene, the radiance registration, and the inverse illumination calculations. The influence of misaligned lightprobes is analysed, as well as the performance of the relighting.

A The synthetic scene set-up

The synthetic scene, shown in figure 6.26, consists of four walls, a ceiling, a floor and one white triangular prism. The ceiling contains an area light source. The four walls have different colours (blue, red, green and orange). The white triangular prism intersects one of the walls and shows considerable colour bleeding from the green, blue and red wall onto the sides of the triangular prism. The geometric model of this synthetic scene is known, and the input images are rendered from known viewpoints. This results in an accurate radiance texture extraction, which in turn allows assessment of the presented methods without the influence from an incorrect geometric reconstruction.

Figure 6.27 (a) shows the same synthetic scene with a wireframe projected over the scene surfaces. This wireframe shows the triangles of the scene. Some triangles are black, they belong to the *neutral cluster*, other triangles have radiance projected onto them. There are no patterned textures. The triangles belonging to the blue wall form one material cluster, the triangles consisting of the white triangular prism also form one material cluster. The BRDF values of these material clusters are known and listed in table 6.1. Image (a) also shows three coloured spheres which represent the positions of the lightprobes used. The lightprobe images are generated using an ideal set-up: an orthogonal projection on an infinitely small sphere with the positions of the lightprobes and their cameras known. The illumination did not change throughout the rendering, therefore the material clusters are also illumination clusters.

	blue MC	white MC
actual	[0.0 0.0 0.6]	[1.0 1.0 1.0]

Table 6.1: *BRDF values for the material clusters in the synthetic scene.*

[3] A demo giving an overview of the developed method using REFLECT can be found online at [JNVL06b].

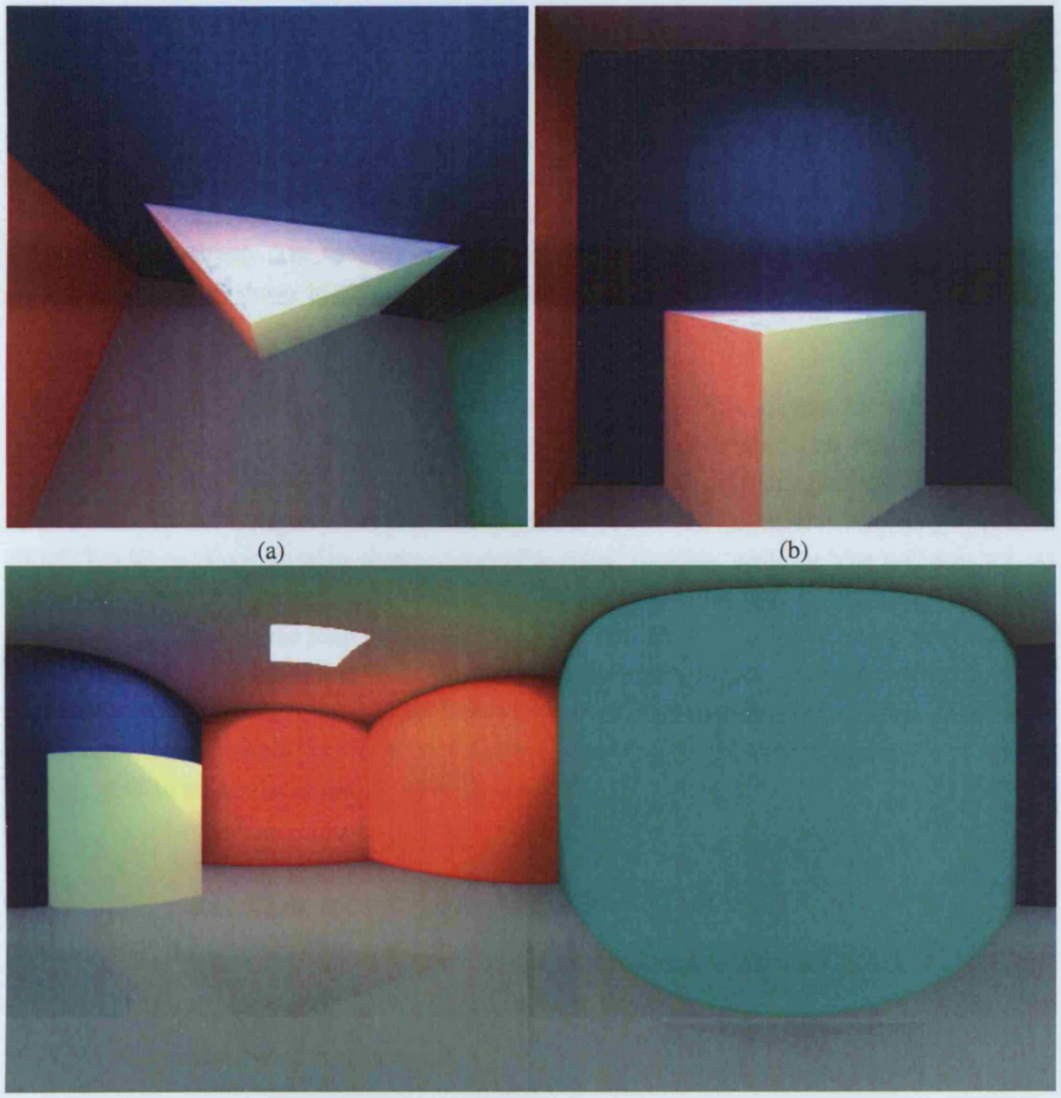


Figure 6.26: *A synthetic scene consisting of four coloured walls (blue, red, green and orange), a white triangular prism, a grey floor, and an area light source in a black ceiling. (a) and (b) show two input images from different viewpoints, (c) is a latitude-longitude image and gives an overview of the entire 3D scene. The white object contains considerable colour bleeding from the red, green and blue walls.*

B Radiance registration

Figure 6.27 (b) and (c) show the back-projection of the lightprobe images of respectively the green and purple lightprobes, shown in (a), onto the 3D scene. The green lightprobe *sees* the entire 3D scene: there is no missing radiance after the back-projection as shown in (b). The back-projection of the purple lightprobe shown in (c) illustrates that some scene surfaces are occluded from the lightprobe. The left side of the white triangular prism does not receive any back-projected radiance, as a result its back-projected texture is completely black. Some parts of the blue wall are occluded from the lightprobe used in (c). The back-projection smears the back-projected radiance values within a triangle. Sometimes this is not sufficient to fill in all missing parts of an illumination cluster. Those missing pixels receive the average radiance of the triangle they belong to. This can result in strange patterns as can be seen in (c): due to quantization effects at the border of the blue MC some radiance from the white MC is smeared into the blue MC. Nevertheless, the other parts of the back-projected textures are perfectly aligned with the underlying geometry. The back-projection of the yellow lightprobe onto the ceiling which contains an area light source is shown in (d).

C Inverse Illumination

After back-projecting the lightprobe images onto the scene geometry, the initial BRDF values are set (incorrectly) to $[0.5, 0.5, 0.5]$. The correct reflectance values are obtained in less than three iterations when using 128 samples per pixel. Figure 6.28 (a) and (b), show the evolution of the BRDF values in the RGB-channel for the triangles in the blue MC (a) and the white MC (b) before calculating a coherent BRDF. The white triangular prism shows considerable colour bleeding, while the blue wall does not. As a result the blue MC converges faster (in one iteration) than the white MC (in two iterations). Table 6.2 gives an overview of the RGB components of the BRDF values per material cluster after applying BRDF coherence. Up to a minor inaccuracy the estimated BRDF values are close to the real BRDF values. As table 6.2 illustrates, the BRDF has an error of $< 1\%$ for the blue and $< 1.3\%$ for the white MC. Furthermore, experiments showed that the error is inversely proportional to the number of samples used per pixel to render the images R_i, as is expected.

	blue MC	white MC
actual	[0.0 0.0 0.6]	[1.0 1.0 1.0]
after 3 iterations	[0.0009 0.0009 0.6093]	[0.995 1.0061 1.0091]

Table 6.2: *BRDF values for the material clusters in the synthetic scene.*

D Influence of misaligned back-projection

To analyse the influence of a misaligned back-projection, due to for instance an incorrect lightprobe or camera position, a small perturbation on the three lightprobes was put through. Each lightprobe is randomly misplaced, the total misplacement is within 10%, relative to the scene dimensions. As a consequence the back-projected textures of the repositioned lightprobe-camera pairs are misaligned, see figure 6.29. It is interesting to analyse the influence of such a misaligned back-projection. It is expected that when the positions of the back-projected light sources are incorrect, the inverse illumination calculations are severely affected. This is because the denominator, used during the BRDF calculations, is a sum of the radiance values, therefore brighter radiance values have a stronger influence than darker radiance values.

Table 6.3 lists the BRDF values of the blue and white material cluster after 3 iterations using 128 samples per pixel. The blue BRDF component of the blue MC is lower ($\approx 3\%$ incorrect) than its actual value, the

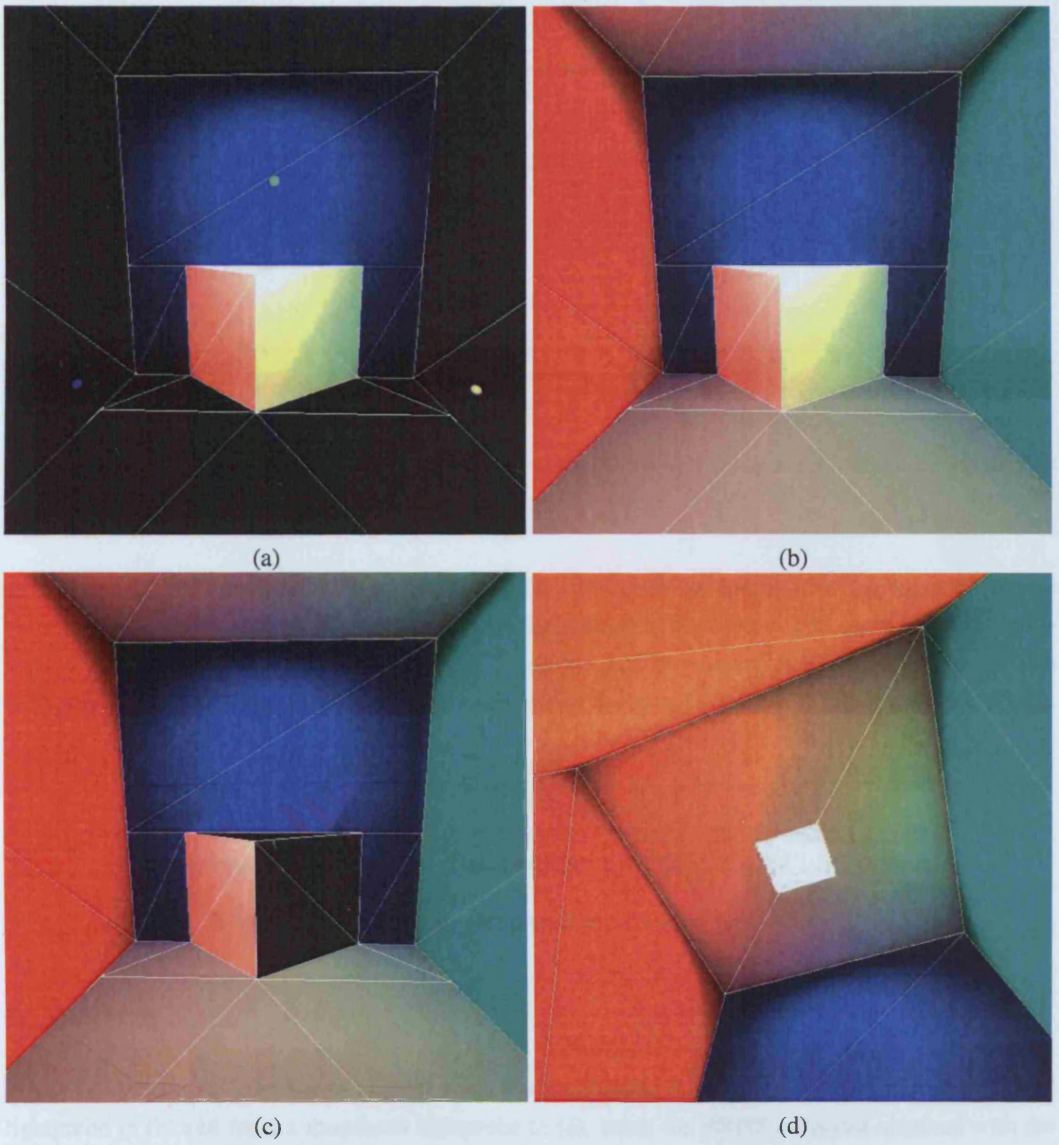
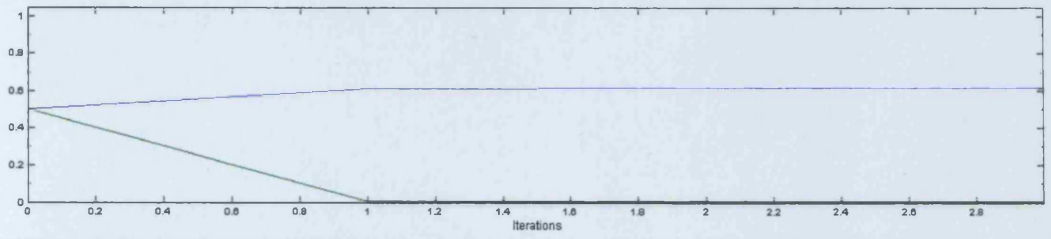
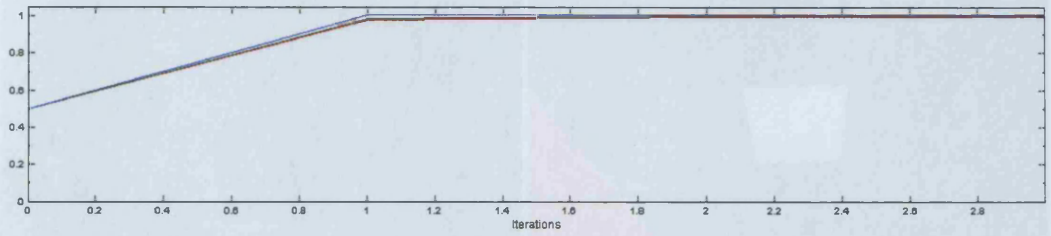


Figure 6.27: *Image (a) shows the triangle mesh of the synthetic scene. The black triangles belong to the neutral cluster, the triangles with a blue texture form one MC, the triangles with the white texture belong to another MC. The small coloured spheres visible in the scene indicate the position of the three lightprobes of the scene. Images (b) and (c) show the back-projected textures of respectively the green and purple lightprobe shown in (a). Image (b) shows the back-projection of a lightprobe that reflects the entire room. Image (c) shows the back-projection of a lightprobe that cannot see the entire 3D scene. In (c) the left face of the white triangular prism remains black as it is not reflected onto the lightprobe. The blue wall shows some smearing effects, this results from the fact that some parts of the blue wall are not visible in the respective lightprobe. This is solved by smearing the radiance of the blue wall into the missing pixels, if this does not fill all missing pixels, these pixels receive the average radiance of the triangle they belong to. Due to quantization effects, some pixels of the white object are smeared into the blue triangles. Besides small local errors, the back-projection is well aligned.*



(a)



(b)

Figure 6.28: *Images (a) and (b) show iteration data for two material clusters: the blue wall (a) and the white object (b). The iteration converges in less than 3 iterations.*

estimated BRDF of the white MC is not pure white ($\approx 10\%$ incorrect) and shows a significant difference between the three components. This is due to the misplacement of the texture. The appearance of *strange* colour components in the BRDF of the white MC results from the incorrect back-projection on the blue and green wall, which results in an insufficient removal of the colour bleeding visible in the white object.

	blue MC	white MC
actual	[0.0 0.0 0.6]	[1.0 1.0 1.0]
after 3 iterations	[0.0009 0.0009 0.6093]	[0.995 1.0061 1.0091]
after 3 iterations, misaligned	[0.0009 0.0007 0.4803]	[0.9164 1.047 0.9214]

Table 6.3: *BRDF values for the material clusters in the synthetic scene.*

E Relighting

Figure 6.30 shows an input image I_i in (a) and a rendered image R_i resulting from an originally aligned lightprobe in (b) and from a misplaced lightprobe in (c), using the BRDF estimates obtained with the aligned lightprobes in (b) and the misaligned lightprobes in (c). While (b) is almost identical to (a), the blue wall in (c) has a different colour compared to that in (b) and (a). This is due to the lower blue component in the estimate for (c) compared to (a) and (b). However, in general the relighting using the misaligned back-projection looks similar to (a). This is due to the fact that the position of the back-projected light source for the misaligned lightprobes, see figure 6.29, is similar to the original position of the light source, see figure 6.27 (d).

Figure 6.30 (d) and (e) show the absolute error between respectively images (a) and (b) and images (a) and (c). We can see that besides the incorrect blue BRDF component, there are also some errors visible in the white triangular prism. These are due to the misaligned lightprobe back-projection which influences the colour bleeding. The maximal relative error (against the original image data) is 1.88% in (b) and 2.93% in (c). The error in (b) is due to inaccuracies in the back-projection, while in (c) due to inaccuracies in the back-projection and the misalignment of the lightprobes.

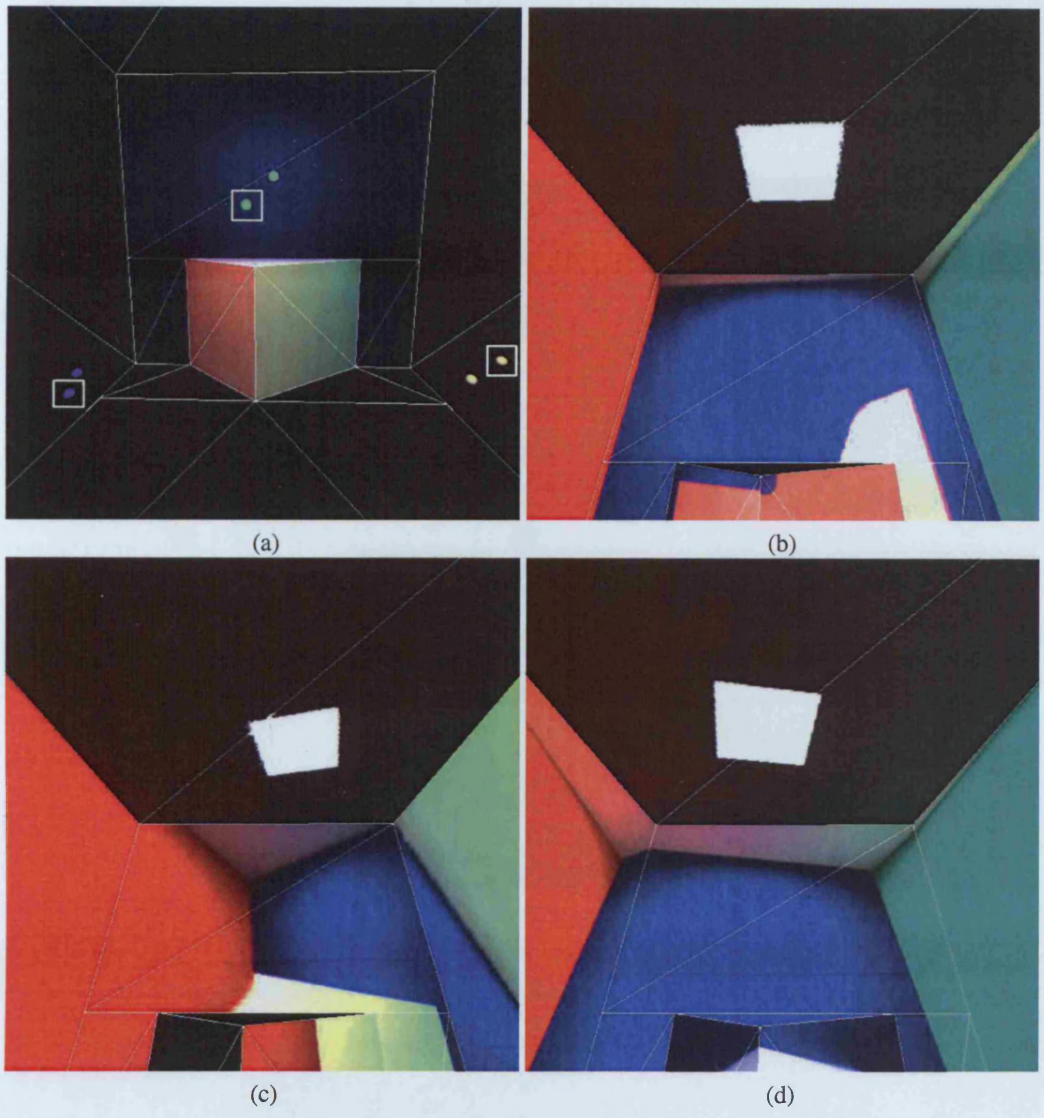


Figure 6.29: *To assess the influence of an incorrect lightprobe position on the inverse illumination calculations, small perturbations of* 10% *misplacement of the lightprobes was put through. As a result the back-projected textures do not align with the scene features. In (a) the boxed spheres are the misplaced lightprobes, the original (un-boxed) lightprobes are shown as well. (b), (c) and (d) show the back-projected textures.*

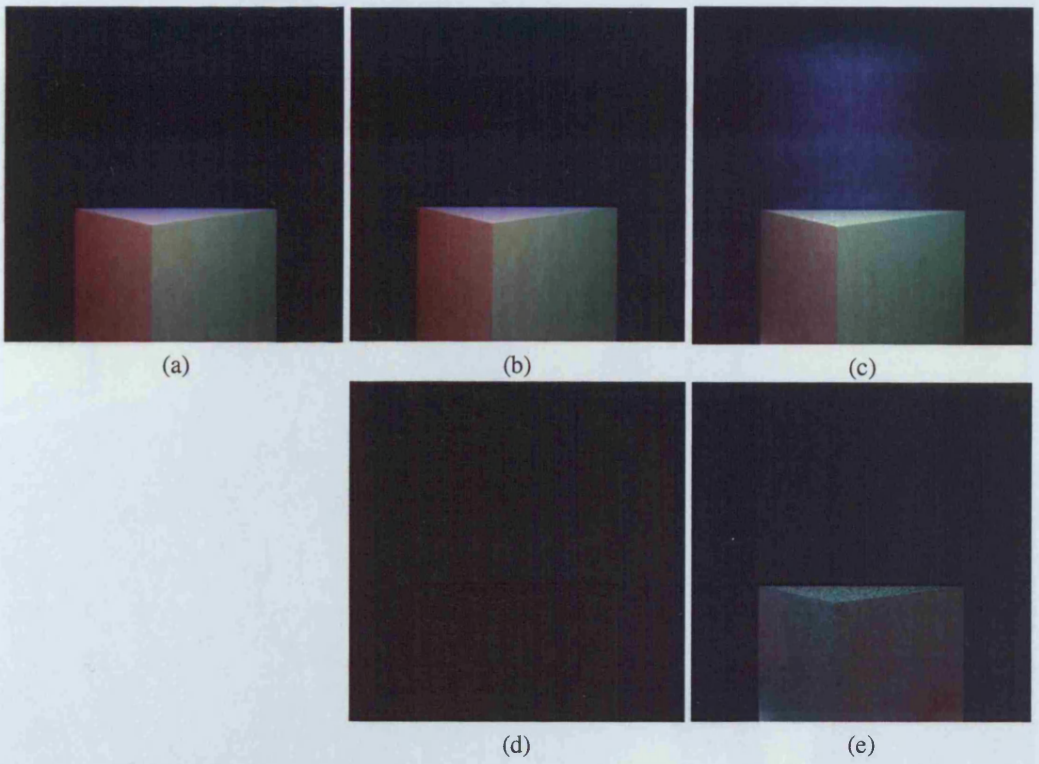


Figure 6.30: *(a) Input image I_i. (b) Rendered image R_i, from the same viewpoint as I_i, and using the BRDF estimates obtained after 3 iteration and applying BRDF coherency. (c) Rendered image R_i, from the same viewpoint as I_i, and using the BRDF estimates obtained after 3 iteration and applying BRDF coherency for the synthetic scene with misaligned lightprobes (see also figure 6.29.). (d) and (e) The absolute difference between respectively (a) and (b), and (a) and (c).*

6.8.2 Results for a real scene

A Real scene set-up

A room containing white walls with several coloured posters displayed on them was modelled using ImageModeler 3.5 [Rea]. A Canon EOS 10D was used to capture the input and lightprobe images. The scene was already presented in figure 6.12, the white wall, the carpet, the coloured cardboard posters and the door form different material clusters. That same figure also showed the different illumination clusters. Different parts of the room were captured under different illumination conditions: the white wall at the right hand side and some parts of the carpet were captured under a different lighting than the other illumination clusters of the room. Figure 6.31 shows four different viewpoints of the textured 3D model. To preserve certain features after the reflectance calculations, the carpet, door, radiator, white-board and projector screen are labelled as *patterned*.

Figure 6.31: *A 3D model of a real scene from four different viewpoints. The texture extraction implemented with ImageModeler introduces boundary problems, this highlights the boundaries between different illumination clusters. Please note that the room shown consists of white walls, which appear yellow in this image due to tonemapping effects.*

The texture extraction implemented in ImageModeler introduces some boundary errors, which result in

visible boundaries between illumination clusters. Ten lightprobes were used to capture the illumination at different places in the scene. The centre of all the lightprobes is $53.75mm$ above the floor. The reflective sphere is positioned on a 22mm high support unit and the diameter of the sphere is 2.5"=63.5mm.

B Lightprobe calibration

To verify the position estimation of the lightprobes, the positions of three lightprobes were modelled in ImageModeler through a marker positioned under the lightprobe. These modelled positions are not used in the selection of corresponding point pairs during the lightprobe calibration, to ensure an objective assessment. Comparing the estimated position with the modelled position does not give an unbiased error measure on the position estimation, as the modelled positions are also prone to errors due to the 3D geometry modelling. As an example, the heights of the 4 corners in the scene with dimensions $\approx 3m \times 4m \times 3m$, have a variance of $5cm$. Therefore, it should be taken into account that the modelled positions of the lightprobes and the selected corresponding point pairs are subject to the same error.

	Lightprobe 1 [cm]			Lightprobe 2 [cm]			Lightprobe 2 [cm]		
	x	y	z	x	y	z	x	y	z
modelled	70.73	-2.02	189.88	80.16	-1.14	63.91	248.31	-13.03	133.58
estimated (6)	74.69	1.70	188.67	83.77	2.08	64.94	246.99	-10.32	132.22
estimated (>6)	72.60	-0.36	189.61	81.08	1.53	64.77	248.38	-12.10	134.04
difference (6)		5.57			4.95			3.31	
difference (> 6)		2.52			2.95			1.04	

Table 6.4: *Estimating the positions of the lightprobes*

Table 6.4 lists the modelled and estimated positions of the three lightprobes. At first a set of 6 points are used, then a few more points are selected to improve the results. The results look promising since with only 6 chosen corresponding points, the error does not exceed 6cm. When more than 6 points are used the error is less than 3cm, which falls below the scene's geometric variance. The total number of selected points never exceeded 20, adding more points did not change the position estimate.

Finding the position of the lightprobe from the lightprobe image is similar to a chicken-egg problem. We need to assume at first that the scene lies at infinity, or in other words we need to work from a distorted latitude-longitude image. During the selection procedure special care was taken not to select points that were most likely subject to significant distortion. This reduces the practicality of the selection procedure. The low resolution of the lightprobe images decreases the quality of the point selection. Further excluding points that might be disturbed by distortion makes the available choices very limited. When more than 6 points are selected, obviously some points might be more subject to distortion than others. However, the results show that even then, a good position estimate can be obtained.

C Radiance registration

The results of the radiance registration are analysed visually. Figure 6.32 shows a set of screen shots of the virtual model with the back-projected textures of several lightprobe images. The left column shows the back-projection without distortion removal, whereas the right column shows the back-projection *after* distortion removal. A wireframe of the scene geometry is projected over the surfaces to allow easy identification of the position of the modelled scene objects. The red contours indicate where the difference in distortion can be best assessed. The images show that the distortion removal improves the alignment between the texture and the reconstructed model. They also show that not all distortion is removed. The reason for the non-perfect distortion removal is twofold: the lightprobe calibration is not perfect and the 3D geometric model is inaccurate. Some triangles are black, this occurs when there

are no points from that triangle visible in the lightprobe image. It is important that this only occurs for insignificant triangles, with low radiance values.

The resolution of the back-projected textures is low compared to the resolution of the original images. Since the back-projected textures are in fact of the same size as the original textures, the final back-projected textures appear blurred. This is not really a problem, as long as the light sources in the scene are relatively large homogenous emitters.

D Inverse illumination

The iterative BRDF estimation usually converges after 3 iterations, when using between 32 to 256 samples per pixel to render the reference images R_i. The experiments showed that while a low number of samples (32-64) returns satisfying BRDF values for the diffuse triangles, usually a high number of samples (64-250) are required to retrieve a satisfying BRDF estimate for the patterned triangles. This is expected because, using the coherency principle the non-patterned diffuse triangles distribute their BRDF estimates across an MC after each rendering R_i, while a patterned MC does not and relies on the quality of the rendering per triangle. Using a lower number of samples reduces the accuracy of the rendered image, therefore also reduces the quality of the BRDF estimate per triangle.

Figure 6.33 (a) shows a rendering of the scene R_i from the same viewpoint as the input image I_i shown in (b). Image (c) shows a tonemapped error image calculated as the relative Root Mean Square (RMS) value between I_i and R_i (the RMS of a pixel divided by the pixel value) and reveals errors in areas containing un-modelled geometry (borders + ventilator above the door), and at un-modelled specular highlights (doorknob). Nevertheless, the maximum error is still low, it is 8% near the edges of the doorknob due to the un-modelled specular effects. The triangular grid is slightly visible in the patterned texture of the door in (a), this effect is due to our per-triangle BRDF update, as explained in section 6.7.1.

Another visible error in figure 6.33 is due to the light source model: the shadow due to the fall-off factor of the light source lies higher in the rendering (a) than in the input image (b). Using the textured area light sources, a light source is modelled as a flat diffuse emitter, while in reality it is a light embedded in a box, surrounded by convex mirrors. As the light distribution is reconstructed using a lightprobe positioned on the ground floor, this light distribution is not correct for points distant to the lightprobe, such as the points on the ceiling.

Figure 6.34 (a) illustrates the estimated BRDF for all scene points. The BRDF is calculated while assuming that all scene surfaces are patterned. The update step is carried out per triangle, and this is the reason why in figure 6.34 (a) triangular artefacts show. For the white wall the BRDF values near the top are darker and of slightly different colour than the BRDF values for the triangles near the ground. The estimates for the triangles near the ground seem more correct (closer to white) as they are closer to the reflective sphere used to calculate the BRDF values. This BRDF per triangle visualisation shows the effects of not modelling all objects (light switch next to the door, ventilator above the door), which influence the BRDF estimate at the location of those missing objects. The darker BRDF near the ceiling is consistent with our conclusion about the incorrect illumination model: the rendered shadow due to the fall-off factor of the light source does not cover all triangles that are actually inside this shadow in the input image. The mismatch in intensity between the rendered image and the input image is compensated for by generating a darker BRDF, which in the next iteration would result in a darker rendered impression. The BRDF seems to be correct for the points and triangles at the top of the wall. Due to the orientation of those points to the light sources, which is almost 90 degrees, they are less dependent on the actual light source model, as they do not receive much light anyway.

Figure 6.32: *The first column shows the back-projection of 3 different lightprobes without distortion removal. The second column shows the back-projection after the distortion removal. After distortion removal, the lightprobe shadow becomes visible in the texture. Not all distortion is removed, due to inaccuracies in the lightprobe position and geometric model. The resolution of the back-projected textures is much lower than the original textures. Sometimes, when a specific triangle is not visible in the lightprobe image, its back-projected texture is entirely black.*

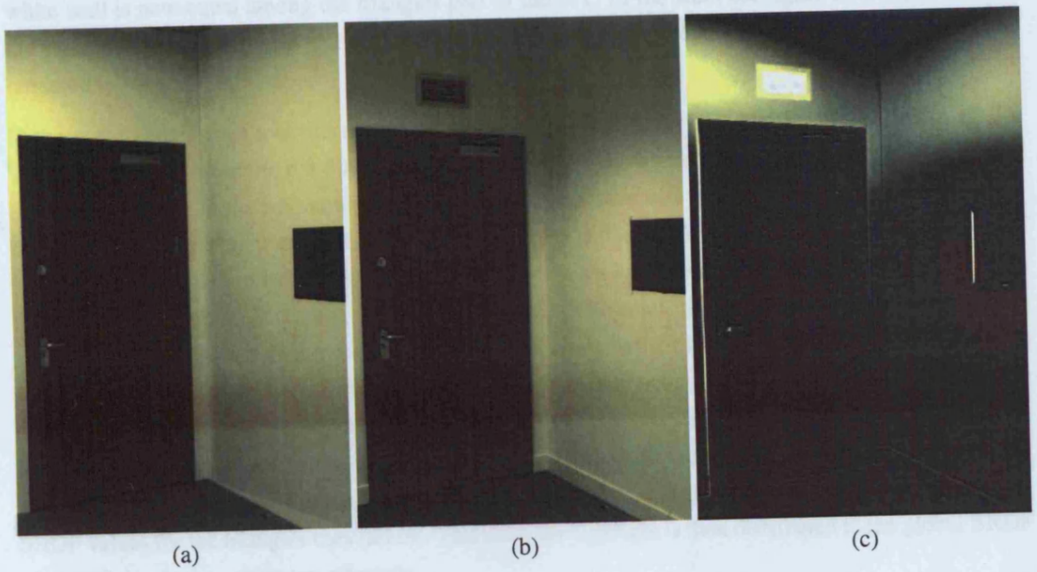


Figure 6.33: *(a) Estimated input image R_i using 2024 samples per pixel, (b) original input image I_i and (c) the error image.*

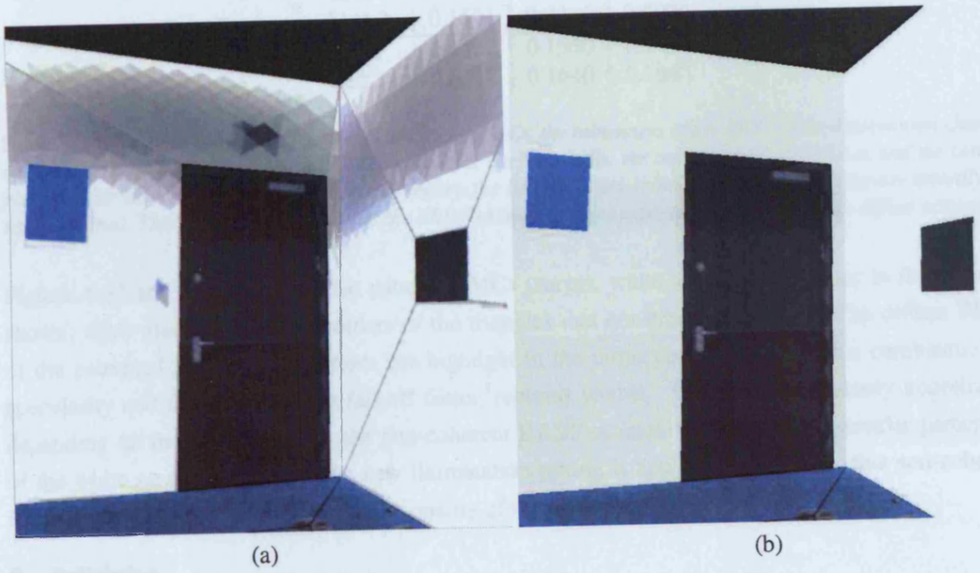


Figure 6.34: *(a) BRDF calculated per pixel, (b) BRDF after applying coherent weighting.*

When we apply the coherency principle to all triangles that are part of the white wall and the blue and green poster, the artefacts due to the incorrect illumination model are removed. The BRDF for the white wall is now equal among the triangles part of the MC of the wall, see figure 6.34 (b). However, the inaccurate illumination model is still perceivable through the fall-off shadow that is visible in the rendered images R.

In the scene used throughout this section, the four walls are actually considered to be different MC's. This enables us to examine the difference between the BRDF calculations when different illumination conditions apply. Table 6.5 shows the BRDF values for three sets of MCs: the four walls, the two brown and the two blue posters. Each was considered a different MC, but each set consists in fact of the same material. Considering that only diffuse components have been estimated, these results confirm that the method provides consistent BRDF estimates. The small difference can be attributed to the small lightprobe misalignments. Some BRDF values overshoot their maximum ([1,1,1]), which indicate something is compromising the BRDF estimate. The most likely contributor to an overshot BRDF is the imperfect compensation of the lightprobe's non-specular behaviour, as discussed in section 6.5.2. If the scaling factor is too low, for instance, the rendered image is always darker than the input image, resulting in an error update to a BRDF value larger than one. Also, specular highlights can enforce artificially high BRDF values for the triangles they fall on. This artificial highlight is then distributed to the global BRDF of an MC due to the coherency principle.

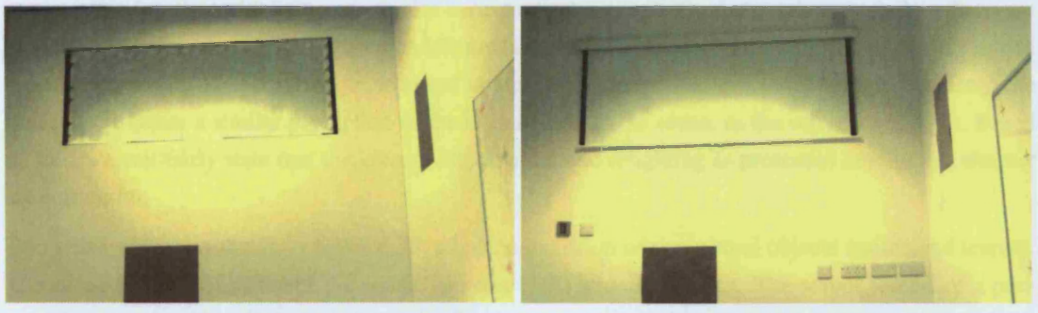
MC	RGB-BRDF		
	R	G	B
wall 1	1.0847	1.0875	1.1170
wall 2	1.1143	1.1054	1.1219
wall 3	0.9645	0.9663	0.9827
wall 4	1.0212	1.0165	1.0837
brown 1	0.4209	0.3298	0.2720
brown 2	0.4609	0.3819	0.3111
blue 1	0.1090	0.2343	0.5673
blue 2	0.1501	0.3218	0.9072
red 1	0.6532	0.1550	0.1445
red 2	0.6318	0.1640	0.1860

Table 6.5: *When considering the four walls as different MCs, the robustness of the BRDF from illumination changes can be assessed. This table shows the BRDF value for the four walls, the two brown, the two blue, and the two red posters, after three iterations. Within each category, the BRDF values should be the same as they are actually the same material. This comparison shows that the BRDF estimates are as expected, considering only diffuse estimation.*

Figures 6.35 and 6.36 illustrate that patterned MCs (carpet, white screen and radiator in the example shown) show artefacts near the borders of the triangles that constitute these MCs. The diffuse BRDF of the patterned MC is not coherent: the highlight in the projection screen, due to a combination of specularity and the light source's fall-off factor, remains visible. This is not necessarily undesirable, depending on the application, as the non-coherent BRDF estimate now creates a specular perception of the white screen. However, if a new illumination setting is applied to the scene, this non-coherent incorrect BRDF estimate will reduce the quality of the relighting.

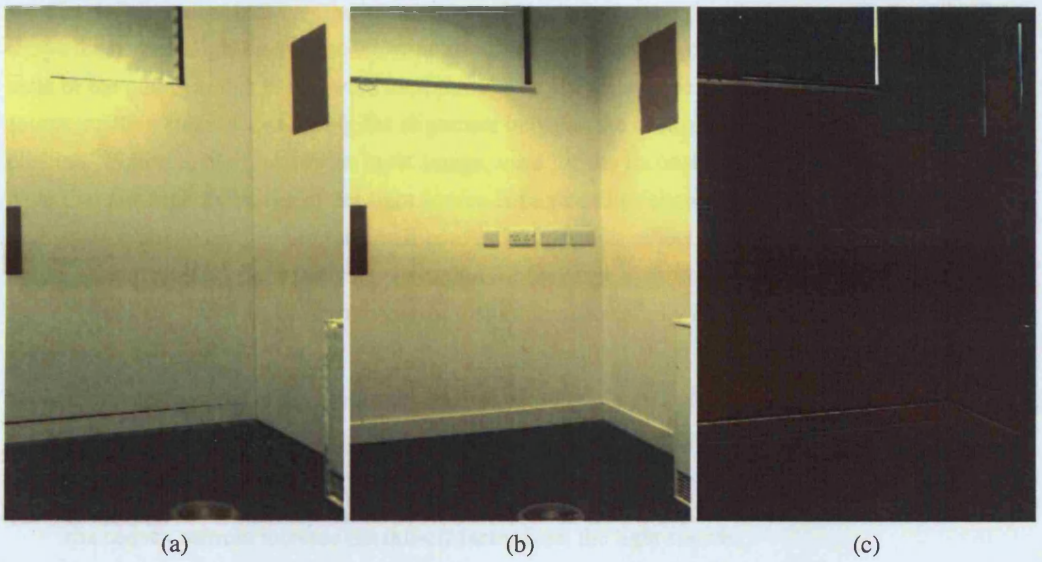
E Relighting

Once the BRDF properties of the 3D reconstructed model are estimated, the scene can be virtually relit with any novel illumination pattern. In our application a novel illumination pattern was captured with a new lightprobe image (LP_r), see figure 6.37 (a). The position of the lightprobe is estimated using the calibration method presented in section 6.5 and its radiance is back-projected using the method



(a) (b)

Figure 6.35: *(a) Rendered image R_i. The white screen shows considerable artefacts due its patterned status. (b) Input image I_i.*



(a) (b) (c)

Figure 6.36: *More results after the inverse illumination iterations. (a) Rendered image R_i. (b) Associated input image I_i. (c) Error image.*

described in section 6.6. To evaluate the quality of the relighting, a comparison image (I_r) is captured under the same illumination as LP_r, and the relighting is carried out from the same viewpoint as the comparison image. Figure 6.37 shows two relit images (b) and (e), from the same viewpoint and under the illumination conditions as the comparison images (c) and (f). (d) illustrates the error on (b) when compared to (c). (b) and (e) are very similar to respectively (c) and (f), except where specular effects are clearly visible, for instance on the door. Additional distinctive artefacts on the side of the door are due to a crude calculation of E per triangle, to speed up the computation. Besides some missing geometry, the relit images create a similar perception of the illumination in the scene, as the reference images. Based on this, we can fairly state that the inverse illumination and relighting as presented here in this chapter are successful.

Two relit images are shown in figure 6.38, after the inclusion of two virtual objects (statue and teapot). The statue consists of different materials: specular (left) and clay (right). The teapot consist of a pure specular material. The shadow effects and reflections associated with the specular objects seem to have realistically simulated.

F Light source model

Applying the coherency principle to the BRDF estimates of the triangles inside a material cluster removes several local errors such as geometric errors, texture inaccuracies, and the inaccuracies in the lighting extraction for points away from a lightprobe.

We discussed how the incorrect illumination model (flat emitter rather than a boxed light source with a fall-off factor) could be corrected for by applying the coherency principle. To analyse the influence of a correct illumination model, we coarsely modelled the light source as a boxed light, where the dimensions of the box are modelled from images. Figure 6.39 (c) illustrates this model: the back-projection of one lightprobe image is shown along with the model's wireframe. From this image we can see that the alignment between the back-projected light source and the model is good. Figure 6.39 (b) shows the original light source model, which consists of a rectangle positioned on the ceiling. Note that none of the points in this model were used throughout the lightprobe calibration, we modelled the light source position solely for verifying the alignment between the back-projection and the real light source position. Figure 6.39 (a) shows an input image, used for the reconstruction of the light source model. Note that although the shape of the light source is modelled as shown in (c), the actual light source is even more complicated, due to the small convex mirrors that reflect the light into different directions. Figure 6.39 (d) and (e) show the back-projection for the same lightprobe images, when distortion effects are not removed. It is clear that removing the distortion considerably improves the alignment of the light sources.

The inverse illumination process seemed virtually unaffected by the difference in light source model and the removal of distortion. We identified two reasons for this unexpected behaviour:

- The light source model is more complex than just a boxed frame around the light source, because the convex mirrors increase the fall-off factor from the light source.
- The distortion introduced by an incorrect back-projection affects other parts of the scene, but to a lesser extent the points in the ceiling, where the light sources are positioned.

The latter can be illustrated by the distortion images shown in figure 6.40 which is similar to the ones shown in figure 6.23. Though the maximum distortion introduced is more than $1m$ in a room with dimensions $\approx 3m \times 3m \times 4m$, this error mainly occurs on the floor plane, and the corners of the room. The error introduced on the ceiling lies around $10cm$.

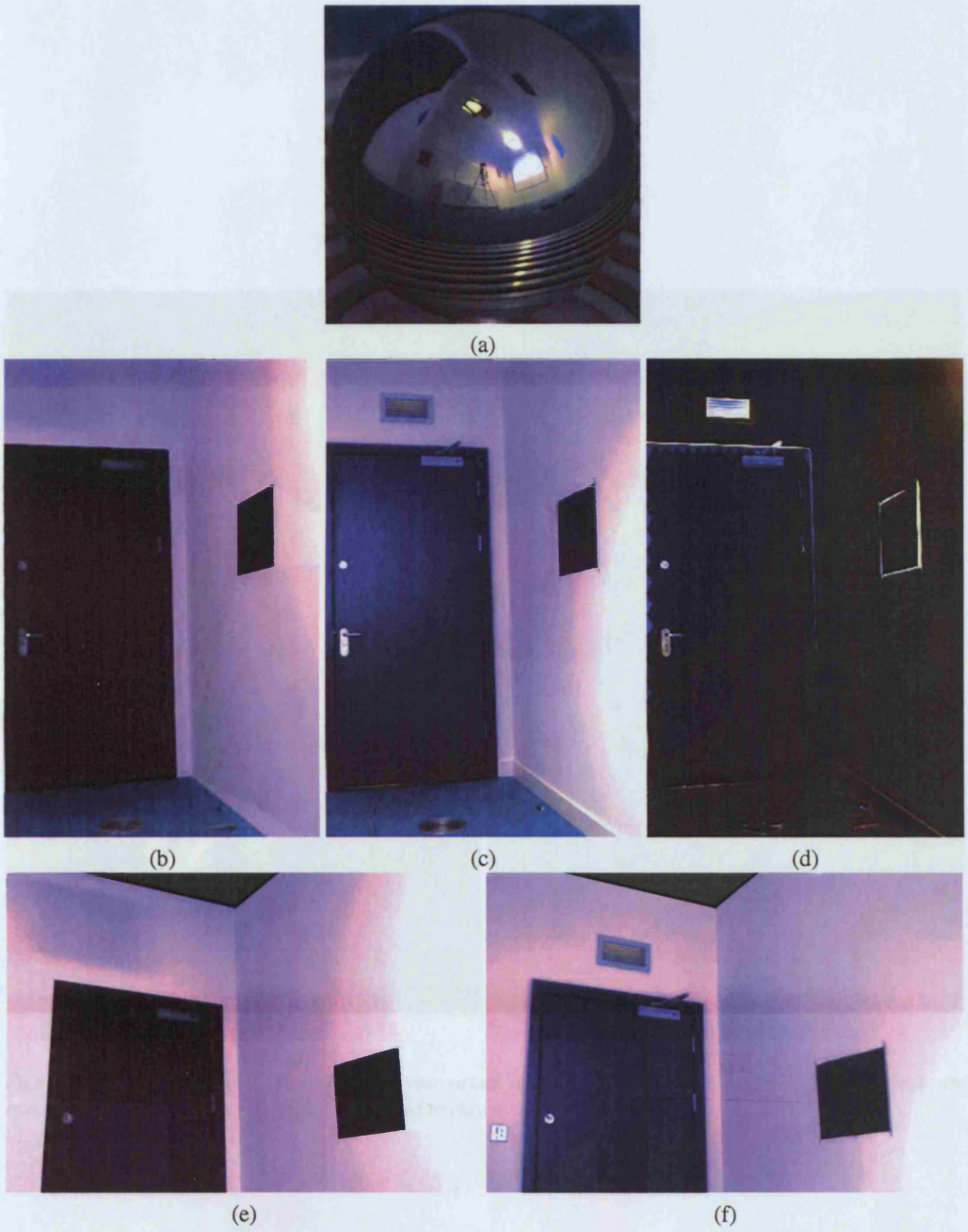
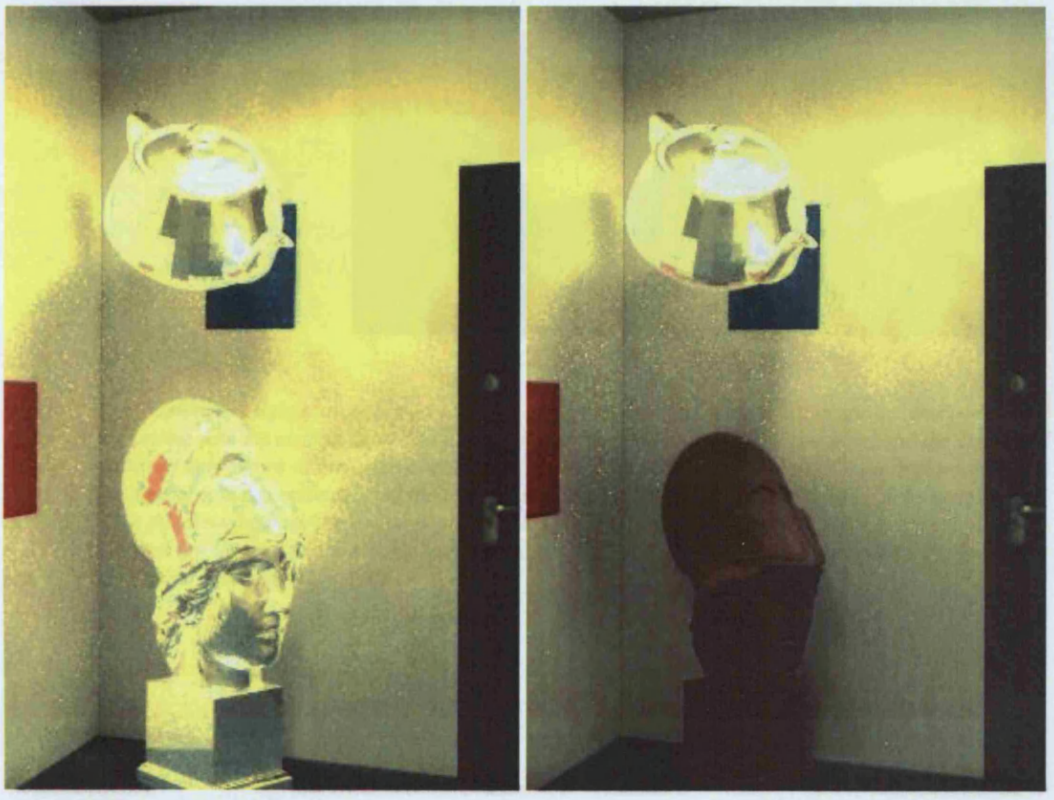


Figure 6.37: *(a) Novel illumination pattern. (b) and (e) Rendered images using 1024 samples per pixel and using the illumination settings defined by the lightprobe shown in (a). (c) and (f) Corresponding input images. (d) Error image for the rendered image shown in (b). The similarity between (b) and (c), and between (e) and (f) is reasonably good, considering only diffuse BRDF values have been used for the relighting. The main difference are due to the unmodelled specular highlights on the door and the doorknob, and the missing ventilator in above the door in (a) which has been removed through the coherency principle.*



(a) (b)

Figure 6.38: *Relighting after the inclusion of two virtual objects: a teapot and a statue. (a) Both objects are specular. (b) The material of the statue is changed to clay.*

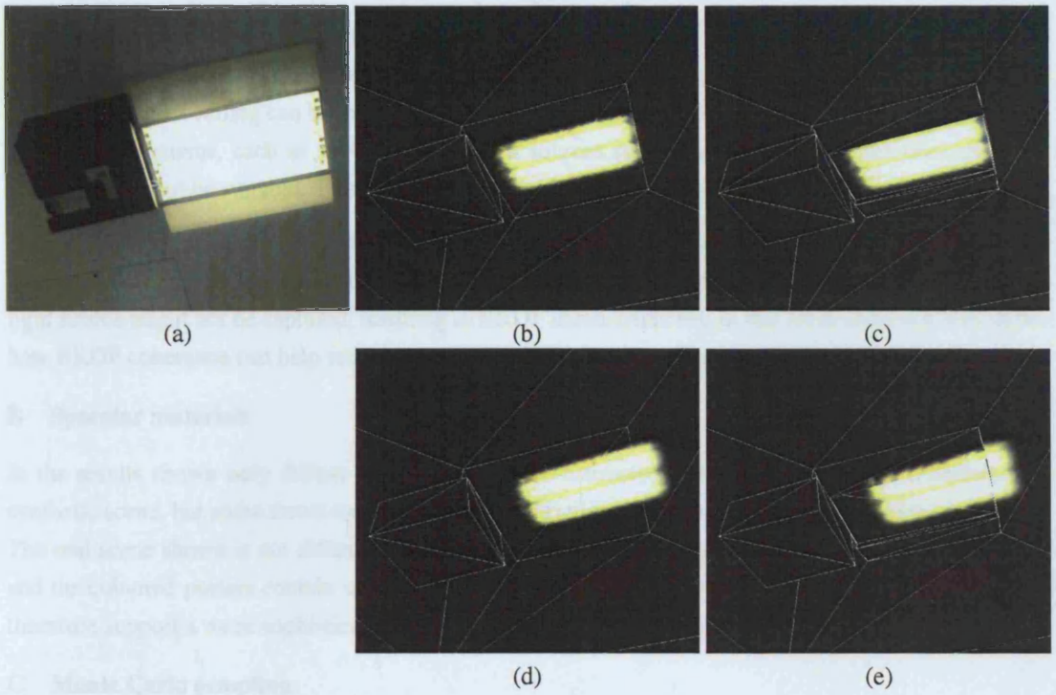


Figure 6.39: *Modelling the light sources in the real scene. (a) An input image of the light source. (b) Back-projected texture on the ceiling with the method described in this chapter. The light source is not modelled and the corners of the light source are only shown to verify the alignment of the back-projected light source with the real position. (c) A coarse light source model is reconstructed from the input images. (d) and (e) show the same model as respectively (b) and (c), but this time the back-projection is performed with distorted latitude-longitude images. In (b), (c), (d) and (e) the images are tonemapped to clearly show the position of the light source in the modelled wireframe, as a result the other parts of the scene are dark.*

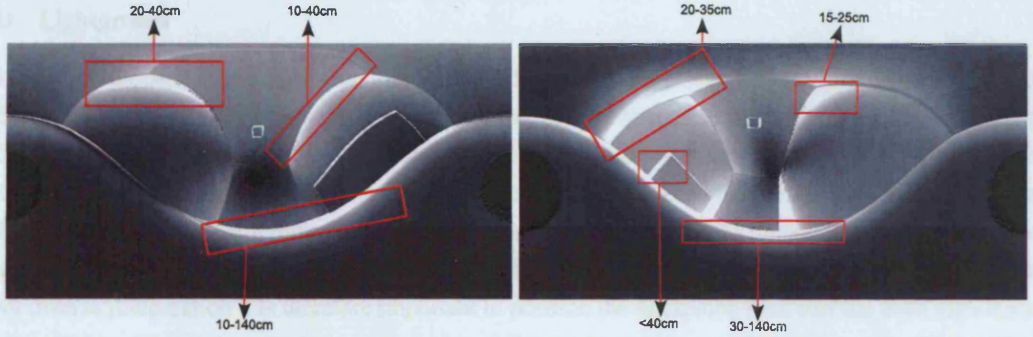


Figure 6.40: *The main distortion introduced by the lightprobes is limited to parts of the scene that do not contain any light source.*

6.9 Limitations and possible improvements

A Light source model

Using textured area light sources to represent the scene radiance of the light sources and un-modelled surfaces in the scene is useful in the contexts shown above. Scenes where the light sources are positioned near a wall or on a ceiling can be modelled using textured area light sources. However, for more complex illumination systems, such as spotlights, and light sources embedded in a box, the textured area light sources might not be suitable. This limits the applicability of the presented method to a certain extent.

As was discussed in section 6.8, when the light sources are near the local scene and if the lightprobes are not positioned near their associated illumination clusters, the local geometric illumination pattern of the light source might not be captured, resulting in BRDF inconsistencies. In that same section it was shown how BRDF coherence can help reducing the inconsistencies due to incorrect illumination models.

B Specular materials

In the results shown only diffuse BRDF values were estimated. This worked well on a diffuse-only synthetic scene, but some errors associated to this restriction were visible in the results of the real scene. The real scene shown is not diffuse-only; the door, the radiator, the ventilators and even the white wall and the coloured posters contain some specular effects. An improvement to our implementation would therefore support a more sophisticated BRDF model, including specular effects.

C Monte Carlo sampling

The path tracer PBRT is slow and produces noisy images. The fewer samples per pixel used, the faster the rendering, but the noisier the image. The drawback of this is that the error used to update the BRDF samples, needs to be calculated per triangle. This creates visually disturbing artefacts, as the triangle borders are visible in the diffuse maps. While the coherency principle removes such local errors, the errors remain visible in patterned material clusters.

The diffuse estimator usually took four iterations to converge, when using about $128 - 256$ samples per pixel, and returned reasonable results. The error images showed relighting errors of less than 10%. Estimating the specular properties of a material is much more computationally demanding. Future work would involve finding an alternative renderer to PBRT. Most likely in the form of a ray tracer, or a radiosity system to retrieve the diffuse BRDF components.

D Lightprobes

Using lightprobes has the advantage that an omni-directional image can be generated from one image. To obtain the same with a fish-eye lens for instance, at least two of such images need to be captured. Using lightprobes has also some disadvantages. First of all, the resolution of a lightprobe image is low. The lightprobe images captured in this chapter never exceeded a resolution of $1900 \times 1900 \times 3$ pixels. The scene in front of the lightprobe covers the main part of the lightprobe image. The remainder of the scene is squeezed into the outer circular band of the lightprobe image. When using the lightprobe images for inverse illumination it is therefore important to position the lightprobe such that the most significant parts of the scene, such as the light sources, are visible in the centre part of the lightprobe. While doing this, the user needs to be careful not to block the light source with the camera, tripod, or his/her own body.

Another problem is the quality of the lightprobe. We have used a chrome-metal ball-bearing. This type of lightprobe is initially coated with an oily lubricant to prevent oxidizing of the outer surface. It

is a tedious work to remove this substance without scratching the ball. Often, small blobs remain on the sphere, giving the sphere an uneven appearance. Though we did not notice any specific reduced performance, the radiance capture can and will be influenced by the reduced specularity of the sphere. Possible future work could develop a pre-processing step to measure the imperfections on the lightprobe, and remove them during the back-projection phase.

E Patterned material clusters

The option to generate BRDF values per pixel preserves certain features in the scene. However, it implies that the coherence post-processing option, which removes local errors, cannot be applied. Therefore errors are more likely to be visible in patterned material clusters than in non-patterned material clusters.

In the results shown in this chapter, it was identified that the update step used to improve the BRDF value is unsuitable for patterned material clusters as the update occurs per triangle, instead of per pixel. This creates aesthetically unattractive errors. An improvement to the current implementation would be a per-pixel update. However, this would require a different type of renderer.

Instead of selecting the patterned option per material cluster, it could have been selected per triangle. This would reduce the total number of triangles labelled as being patterned, which in turn would reduce the visual errors. However, the user can select the patterned MCs at run-time, which is fairly fast. If the user had to select at triangle level instead of at MC level, the manual interaction time would rise considerably.

6.10 Chapter summary

This chapter highlighted the problems that arise in the radiance capture of a scene that is lit with dynamic illumination. When capturing the scene from a set of photographs, illumination changes may appear in the textures extracted from the images. When capturing the scene radiance from such a set of incoherent images, the radiance equations cannot be reconstructed, which limits the use of inverse illumination for BRDF reflectance calculation.

Most existing illumination methods, ignore the problems posed by illumination changes and only operate on scenes that strictly avoid any type of illumination changes. This chapter challenged this problem, and developed a methodology for relighting applications that allow illumination changes to occur during the image capture. The scene radiance is reconstructed from uncalibrated lightprobe images of a reflective sphere. The first contribution of this chapter provided a method to calibrate lightprobe images for accurate radiance registration, even with the constraint that the original lightprobe position is unknown and that the scene lies at a finite distance, near the lightprobe. The novel calibration method is based on a set of minimum 6 point pairs between the lightprobe image and the 3D scene. A novel back-projection method uses the positions of the sphere, the camera, and the 3D scene geometry to calculate the radiance of the scene points visible from the centre of the sphere by solving a fourth degree self-inversive polynomial rather than using ray tracing. This results in a more accurate back-projection compared to the methods that ignore distortion and pinching effects. This accurate back-projection method is useful for several applications, such as relighting and environment mapping.

The second contribution is the application of the radiance capture to inverse illumination and relighting of scenes originally captured under varying illumination. The radiance values captured from the lightprobe image are back-projected onto the scene geometry and are then used to gather irradiance at points near the position of the reflective sphere. This allows the estimation of the reflectance values for surfaces

near the spheres. The results showed that the diffuse BRDF of the materials in a scene can be accurately estimated, and that high-quality relightings of a scene can be generated. A coherency principle was introduced to obtain homogenous BRDF estimates for points belonging to the same material. The coherency principle also removes local errors due to improper geometry and light source reconstruction.

The combinations of these methods provides a relighting algorithm, for scenes captured under dynamic conditions, and reconstructed using user-aided reconstruction software. Some limitations of the developed method were discussed and some solutions advised. The main limitations identified with the current implementation is the Monte Carlo Path Tracer used to render the scene, which due to its poor performance, is unsuitable for an upgrade of the system to a specular BRDF estimator.

Several areas of future work were indicated. Firstly, more research should be carried out on light source modelling. While it was shown that our coherency principle removed errors due to an incorrect illumination model, a better illumination model could improve the BRDF calculations. Secondly, using lightprobes for lighting extraction is an inexpensive solution. However, local imperfections on the reflective spheres, due to greasy fingerprints or erosion effects, can influence the lighting extraction. A pre-processing step could analyse these imperfections and remove them during the lighting extraction phase.

Chapter 7

Conclusion

7.1 Introduction

This chapter summarizes the objectives and contributions of this thesis and provides directions for future work. Section 7.2 revisits the problems identified with simulating illumination for mixed reality of complex-to-model scenes. Section 7.3 discusses the objectives set out in the introductory chapter, and summarizes the contributions of this thesis. Section 7.4 provides an overview of the limitations of the presented methods, and provides several areas of improvement. Finally, section 7.5 gives a conclusion and an overview of the future work that could follow up this thesis.

7.2 Problems revisited

Illumination methods for mixed reality require the geometry and scene radiance to be known. The 3D geometry of a scene can be captured using a scanner, or user-aided reconstruction software. Both methods have their flaws and experience problems when applied to uncontrollable, complex-to-model scenarios. It has been explained in chapter 4 that the length of the capture, the deformation of uncontrollable objects, the complexity of the shapes and the vibrations of the camera can all contribute to an unreliable geometric model. When lighting effects need to be removed from or detected in a radiance texture, the accuracy of the calibrated scene (geometric model and light source positions) plays an important role. We demonstrated that, when not used in the context of illumination methods, an accurate 3D model might not be required for aesthetical reasons as long as there is an appropriate texture mapped onto the geometry. Therefore the efforts to improve the accuracy of the geometric model can be saved, if appropriate illumination methods are developed.

The scene radiance is not only required to represent the illumination in the scene, but also to correctly retrieve the lighting effects, such as glossy highlights and shadows. While inverse illumination uses the scene radiance to calculate the material properties of the scene, common illumination and relighting use the scene radiance to analyse the current lighting effects and how to change them. The scene radiance is usually captured from photographs. Using conventional LDRIs introduces unwanted saturation effects, which make them unsuitable for inverse illumination. A solution for this has been offered by means of HDRIs. HDRIs are images with a wider dynamic range than LDRIs, and do not contain any saturation effects. These HDRIs are generated from LDRIs captured with different exposure settings, or using an expensive HDR camera. Similarly to geometric reconstruction tools, both HDRI generation methods are sensitive to camera movements and object deformations, as was explained in chapters 2 and 5. As a consequence, the radiance capture of scenes affected by movement (public places, outside scenes) can be unreliable and error-prone.

When performing inverse illumination, it is important to match the scene radiance with the scene geometry correctly, and to have a coherent radiance distribution: the scene radiance should be captured under the same illumination conditions throughout the capture process. This is a condition often assumed to be obeyed at all times, or at least easy to obtain by the illumination methods. However, when capturing the radiance of a scene lit by a dynamic light source (e.g., outdoor scene, public spaces, indoor rooms with large windows) this condition of illumination consistency is difficult to obtain, as was illustrated with several examples in chapter 6. The long time required to capture the radiance, further complicates the illumination consistency. As a result, the applicability of illumination methods to real-world scenarios is limited.

From this discussion it follows that the input data required by the illumination methods is often unreliable, which compromises the applicability of these methods to real-world scenarios. This thesis assessed the errors introduced by the incorrect assumption that the geometry and radiance are accurately known and found several solutions for the problems identified.

7.3 Objectives and contributions revisited

Section 1.4 defines the hypotheses of this thesis. The first hypothesis states that the illumination methods are robust against inaccuracies resulting from complex-to-model scenes. This first hypothesis, was falsified through chapters 2 and 3. Chapter 2 explained that an inaccurate geometry and radiance capture may create undesirable artefacts. Moreover, chapter 3 illustrated that the state of the art in illumination methods does not provide any solutions for inaccurate geometry or radiance captures, but will ignore the possible artefacts.

The second hypothesis states that low-cost solutions can be provided to make the illumination methods robust against the inaccurate input capture. Chapters 4, 5 and 6 successfully presented such low-cost solutions. None of the presented methods required special, expensive equipment. All methods can be carried out using a conventional camera, a reflective sphere, and reconstruction software. The presented methods are semi-automatic and require limited user inter-action, such as the selection of 6 corresponding points between lightprobe image and scene geometry to aid the lightprobe calibration.

An overview of the contributions is given in the remainder of this section, starting with the classification, followed by the solutions for the inaccurate geometric reconstruction, and finally the solutions for the inaccurate radiance capture. The relation of each of these contributions to the hypothesis given in the introductory chapter is explained.

A Classification of illumination methods based on input data

Chapter 3 presented a classification of the illumination methods based on the required input data. The resulting classification consisted of four different groups, with the amount of input data required increasing from group one to group four. Several other characteristics could be associated with each group. As an example, the quality of the generated illumination is often proportional to the amount of data available. Similarly, the flexibility of the method decreases with the amount of data required.

In the literature very little information is available about the problems attributed to incorrect geometric reconstructions and radiance captures. Usually the illumination methods avoid such situations. Nevertheless chapter 2 explained that an inaccurate geometry and radiance capture will influence the illumination simulations, which makes the illumination methods sensitive to errors when used in complex-to-model scenes.

This encourages us to find robust, low-cost solutions to compensate for the problems identified in the state of the art of illumination methods.

B Common illumination for inaccurately reconstructed scenes

Common illumination simulates the virtual lighting effects induced by the virtual object in the scene. Common illumination methods further differ in the type of lighting effects generated. These can be local, global, plain shadows, or include global inter-reflections. Most common illumination methods operate in a similar fashion: the virtual lighting effects are simply added over the captured scene radiance. For shadow generation this means that the radiance values within a virtual shadow are scaled to resemble the reduced incidence of light.

The original real scene radiance contains lighting effects associated with the geometry and light sources present in the real scene. Some virtual lighting effects should not be simulated when a similar effect is already present in the real scene radiance. For instance, a virtual shadow simulated by scaling the scene radiance, should not scale the radiance of scene points that already lie inside a real shadow. In several methods, the geometric model is used to retrieve information about the lighting effects visible in the real radiance. However, when the geometric model is inaccurate, the location of the real lighting effects is usually not correctly retrieved. It was illustrated in chapter 4 how this may result in artefacts after the virtual shadow simulation.

A solution has been presented in chapter 4 in the form of a novel framework to generate consistent shadows for mixed reality of poorly reconstructed scenes. The framework consists of the following three steps:

1. Shadow detection: the real shadow pixels, visible in the scene radiance, are detected based on the reconstructed scene geometry, light source position, and the pixel intensities in the texture.
2. Shadow protection: the points inside a real shadow are protected through the creation of a shadow mask. This shadow mask defines if and how the underlying scene points can be manipulated when a virtual shadow would overlap with that scene point. More practically, the shadow mask provides a scaling factor that relates the radiance of a point in shadow with that not in shadow.
3. Shadow generation: The virtual shadows are created by scaling the radiance of the points inside a virtual shadow according to the value define by the shadow mask.

In chapter 4, shadow volumes are used in combination with the above described methodology to generate consistent virtual shadows. The robustness of the method to the inaccuracy of the geometric model and the positioning of the light source has been demonstrated in that same chapter. The restriction to semi-soft shadows is entirely due to practical reasons and the framework as defined above could be adapted to work with soft shadows too, if a good soft-shadow detector exists.

This framework does not have to be restricted to common illumination only. In fact it can be applied to any type of illumination method that requires the position of the lighting effects in the real scene. Re-lighting methods, for instance, require knowledge about the lighting effects in order to correctly remove them prior to applying new lighting effects. The 3-step methodology discussed in this section can easily be adapted for relighting methods.

C Inverse illumination for inaccurate reconstructed scenes

The previous subsection pointed out the need for an accurate geometric model for common illumination and relighting methods. The same holds for inverse illumination, as it uses the inverse of the radiance

equation to calculate the material properties in the scene. To obtain the BRDF of a shaded scene point, it is therefore important to register this point as being in shadow, otherwise the BRDF retrieval is incorrect. While the methodology summarized in the previous subsection can be used to retrieve the position of the lighting effects in the scene, the inaccurate geometry can also be compensated for through a coherency principle.

The coherency principle states that the BRDF of a material should be roughly the same for points belonging the same material. Therefore the BRDF can be reconstructed from the BRDF estimates of all scene points belonging to the same material for which it is most likely that the estimate is correct. The BRDF is calculated using a weighted average, where the weights should reflect the inaccuracy expected on the BRDF estimate. In chapter 6, the inaccuracy is mainly attributed to the distance of a scene point to the lightprobe used to reconstruct the radiance distribution around that point. Retrieving the BRDF as a weighted average reduces the sensitivity of the BRDF to inaccuracies in the geometry and radiance capture.

The coherency principle improves the state of the art in physically-based illumination considerably, as it allows inverse illumination of scenes for which the geometric reconstruction is inexact and the light source model complicated. Through the capture of the scene radiance with a lightprobe, the light source can be modelled as a textured area light source. The reconstructed radiance distribution is only correct for points near the lightprobe, and that is why the coherency principle will improve the global BRDF estimate for point further away from the lightprobe.

D Radiance capture in dynamic scenes

The radiance of a scene can be captured with HDRIs. After camera calibration, the radiance captured in an HDRI can be mapped onto the scene geometry. HDRIs are captured with an HDR camera, or generated from LDRIs captured with different exposures. As discussed in chapter 5 both methods are sensitive to object and camera movement. Both types of movement result in unwanted duplications or ghosting effects. This ghosting effect is due to the weighting of the LDRIs into the HDRI, when generating HDRIs using multiple exposures. The ghosting effects are undesirable, as it means that the resulting radiance values are unreliable estimates of the true radiance of the scene. In the literature limited solutions have been presented to deal with camera movement and even to a lesser extent with object movement. Nevertheless, to proof our second hypothesis, it is important to solve the problems associated with the HDRI capture in complex-to-model scenes.

In chapter 5 the problems of camera and object movement were tackled. First the current method to remove the camera movement is improved by incorporating rotational misalignment effects. Then a novel object movement removal method is developed that has the interesting feature of detecting movement prior to the calculation of the camera curve. This has the advantage that pixels affected by movement can be ignored during the camera calibration.

The resulting method generates HDRIs from a set of LDRIs captured with a hand-held camera and showing object movement. The presented framework is independent from the camera curve calibration, and therefore can be fitted into other HDRI generation methods as a separate module.

The novel HDRI generation model as presented in chapter 5 enables the radiance capture of uncontrollable scenes. The radiance is required in common illumination, relighting, and inverse illumination. The presented automatic, low-cost solution fits well in the objectives of this thesis. It increases the robustness of the illumination methods for mixed reality to uncontrollable scenes, without requiring extra equipment. On the contrary, the presented method removes the requirement of a tripod throughout the

radiance capture.

E Radiance capture for relighting dynamically lit scenes

To calculate the BRDF using inverse illumination, the scene radiance (scene points including light sources) needs to be coherent. When capturing a scene using HDRIs, the length of the HDRI capture can compromise the above given requirement of radiance coherency, especially when capturing an outdoor scene or an indoor scene affected by dynamic illumination.

In the literature this is resolved by capturing the radiance using a reflective sphere. These lightprobes capture the entire 3D scene as seen from the centre of the lightprobe at the exception of the conical volume behind the sphere, relative to the camera used to capture the lightprobe image. The methods in the literature usually record the position of the lightprobe or calculate it using a calibration board, which makes the scene radiance capture more difficult. Furthermore, when back-projecting the lightprobe images onto the scene, the conical volume behind a lightprobe image is usually ignored as well as the finite distance between the scene and the sphere, which results in distortion. This distortion will map radiance values to incorrect scene points, which compromises the robustness of the illumination methods.

In chapter 6, a novel methodology was presented to capture the radiance of scenes subject to dynamic illumination using lightprobes. A semi-automatic calibration algorithm has been developed that uses corresponding points between the scene and the lightprobe images to calibrate the lightprobes in the 3D scene. The radiance of the lightprobe images is mapped onto the scene in a distortion free manner. For this purpose, the software system REFLECT has been developed, which provides an interface to aid the lightprobe calibration, the back-projection of the lightprobe radiance onto the scene, the inverse illumination calculations and finally the relighting of a 3D scene. The system specifications are given in appendix E.

The automatic lightprobe calibration improves the radiance capture, as it allows a faster radiance capture in general. The distortion-free back-projection ensures that the light sources in the scene can be accurately reconstructed from a lightprobe image. The use of reflective spheres to reconstruct the radiance distribution, is a cheap alternative compared to using a fisheye lens.

F The combination of the contributions

The methodologies summarized in subsections B through E can be integrated and used in one illumination method. Although each of these methods has been tested for a specific type of illumination, they are in fact general enough to be used for common illumination, relighting, or physically-based illumination.

For instance, all illumination methods require radiance information. Using the HDRI generation method presented in chapter 5 the radiance can be captured using a hand-held camera in scenes containing uncontrollable objects.

If the geometric reconstruction is inaccurate, the methodologies presented in chapter 4 and 6 can be used to reduce the artefacts that such inaccuracies introduce. The two methodologies can also be integrated. For instance, the coherency principle used to remove geometrical errors, cannot be applied to per-pixel BRDF estimates. Therefore, scene points for which a per-pixel BRDF estimate is required, rather than a global BRDF, are still subject to errors due to the inaccurate geometric model. The methodology from chapter 4 can be used to reduce the errors by pre-detecting the lighting effects in the texture.

For all three types of illumination methods it is important to know the light sources in the scene. The light sources need to be known either to simulate the lighting effects or to retrieve the existing lighting effects. With the radiance capture methodology presented in chapter 6 the information of the light sources can be

retrieved from lightprobe images, regardless of the complexity of the light source under consideration, and without distortion.

7.4 Limitations and directions for improvement

The previous section summarized the contributions of this thesis. The objectives were appropriately met, and several problems with geometric and radiance inaccuracies for illumination methods have been highlighted and resolved. Nevertheless there are still areas within this domain that require further attention. This section lists the shortcomings of the presented methods and provides some directions for future work. First the issues related to geometric inaccuracies are given, followed by those related to the radiance capture.

A Geometric inaccuracies

The method presented in chapter 4 solved the problems related to inaccurate geometric reconstructions in illumination methods. The virtual shadow generation method offers an improvement compared to the available methods within its field. The applicability of the method could be further improved if a soft shadow detector was used instead of the semi-soft shadow detector that is used in our implementation.

The current implementation to derive the scaling factor as the ratio between pixel intensities inside and outside a shadow is a simple and satisfactory solution for the applications used in chapter 4. If more complicated light sources are introduced, such as large area light sources, the scaling factors retrieved as mentioned become erroneous as the real shadow will not have a homogeneous appearance with only umbra, but will consist of large penumbra regions too. Further research would improve the scaling factor retrieval for soft shadows or scenes with more than one light source.

The presented methodology in chapter 4 can be used to track a light source. Once the real shadow is known, the light source position can be updated using the real shadow position and the reconstructed geometry. This could be beneficial for AR applications that operate in a scene lit by a dynamic light source.

The relighting algorithm presented in chapter 6 makes use of a coherency principle to remove errors due to geometrical inaccuracies. The BRDF is initially calculated per triangle. The final BRDF is calculated as a weighted average over the per-triangle estimates. The consequence of applying the coherency principle is that all scene surfaces associated with a certain material receive a uniform BRDF. Local material differences, due to dirt for instance, are removed, which is not always desirable. As a solution we allowed certain triangles to receive a per-pixel BRDF estimate. Due to our choice of renderer, this sometimes resulted in an inaccurate, aesthetically unattractive looking BRDF distribution for neighbouring triangles. An update to another type of renderer, such as a ray-tracer instead of a Monte Carlo path tracer, may remove this error.

In the relighting method presented, the light sources were modelled as textured area light sources. This resulted in errors in the BRDF calculation, as the real light source model is in fact far more complex. Though the coherency principle partially solves the associated problems, the developed relighting method could be improved if the user could select different models of light sources for the inverse illumination calculations, although this would compromise our statement that the illumination is extracted from a lightprobe image, without modelling the light source.

B Radiance capture

The HDRI generation method, presented in chapter 5, reduces the calibration to an Euclidean transformation, while in general the transformation will be a perspective transformation. The alignment algorithm fails when a too large portion of the image is corrupted by a moving object. It was argued that if alignment had been successful, too large a portion of the image would have required a substitution from one exposure, which would have made the final HDRI unreliable. Nevertheless it is inappropriate to dismiss this particular problem, as some applications might want to retrieve the radiance of a large object that cannot be kept fixed. For instance, if the texture of a human is captured to be mapped onto an avatar. The small movements this person makes result in an unsuitable HDR texture. Future research could involve finding a warping function between the pixels of the different LDRIs. This warping function can then be used to compose the radiance information in the LDRIs correctly for the moving object. While several advances have already been made in this area, the existing methods lack robustness against the change in exposure, and in low-contrast areas. A non-feature based warping method based on mutual information will most likely be the direction that this research should take.

The calibration of a lightprobe inside a 3D scene requires manual input. Though the manual input is limited to finding a minimal of 6 corresponding points, it would be interesting to investigate the possibilities of an automatic alignment of the lightprobe into the scene.

7.5 Future work

The focus of this thesis lies on the robustness of the illumination methods to inaccurate geometry and radiance capture. The problems identified resulted directly from the inefficiency of the capture tools to acquire the input of the illumination methods. The sustainability of the presented methods within their research field depend on how well the research into the capture tools improves. With the increasing attention that HDR imaging and 3D reconstruction receives, it is expected that the problems associated to the inaccurate scene geometric reconstruction and radiance capture will diminish.

The focus of future research can therefore be the capture of HDR video, to enable real-time relighting. This would require real-time reconstruction and real-time reflectance estimation. This thesis fits well within these long-term goals of the future work. For instance, the lightprobe calibration and radiance registration presented, when fully automated, is an excellent means to enable real-time capture of the scene illumination required for the reflectance calculations and illumination simulations. Real-time reflectance calculations can be achieved through reducing the search space for the reflectance values in the scene, for instance, by using a database of common real-world reflectance values. The reflectance database can be build through measuring real-world materials with image-based sampling methods such as a goniometer. The methods presented in this thesis to compensate for inaccurate geometry can be combined with real-time reconstruction methods such as structure from motion.

The incentive for this thesis lay on making illumination methods more robust, in order for them to become part of our daily life. To achieve this, not only the inaccuracies related to the radiance and geometry capture tools need to be removed. The accessibility to mixed reality applications must improve too. Currently few people use mixed reality or augmented reality during their daily life activities. The existing applications, such as used in the movies or at museums, are usually designed for large groups of people, following a strict set of tasks, and are usually not robust enough to deal with unexpected user-interaction. This is partially due to the limited robustness of the applications to real-life environments, but also due to the cost often associated with mixed reality systems. This thesis made the first step towards

low-cost solutions to increase the robustness of the illumination methods. However, more research is required to make the developed methods operate in real-time, without user-interaction.

Bibliography

[3rd] 3rdtech. www.3rdtech.com.

[AA01] M. Aggarwal and N. Ahuja. High dynamic range panoramic imaging. In *proceedings of Eighth IEEE International Conference on Computer Vision (ICCV)*, 2001.

[ABB+01] Ronald Azuma, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, and Blair MacIntyre. Recent advances in augmented reality. *IEEE Computer Graphics Applications*, 21(6):34–47, 2001.

[Ado] Adobe. Photoshop cs2. www.adobe.com/products/photoshop.

[ALCS03] Kusuma Agusanto, Li Li, Zhu Chuangui, and Ng Wan Sing. Photorealistic rendering for augmented reality using environment illumination. In *proceedings of IEEE/ACM International Symposium on Augmented and Mixed Reality (ISMAR '03)*, volume 21, pages 208–216, October 2003.

[Any] Anyhere software. Photosphere. http://www.anyhere.com.

[Arv95] James Richard Arvo. *Analytic methods for simulated light transport.* PhD thesis, Yale University, 1995.

[AS00] Michael Ashikhmin and Peter Shirley. An anisotropic Phong BRDF model. *Journal of Graphics Tools: JGT*, 5(2):25–32, 2000.

[ATI02] ATI. Car paint, 2002. www.ati.com/developer/demos/r9700.html.

[Azu95] Ronald Azuma. A survey of augmented reality. In *ACM Siggraph '95, Course Notes ♯9: Developing Advanced Virtual Reality Applications)*, pages 1–38, August 1995.

[BAS] BASLER Vision Technology. Basler a600-hdr. www.baslerweb.com/produkte/produkte_en_1455.php.

[Bev03] Alessandro Bevilacqua. Effective shadow detection in traffic monitoring applications. *WSCG*, 10(2):57–64, 2003.

[BG01] Samuel Boivin and Andre Gagalowicz. Image-based rendering of diffuse, specular and glossy surfaces from a single image. In *proceedings ACM Siggraph '01 (Computer Graphics)*, pages 107–116. ACM Press, 2001.

[BGWK03] Oliver Bimber, Anselm Grundheimer, Gordon Wetzstein, and Sebastian Knodel. Consistent illumination within optical see-through augmented environments. In *proceedings of IEEE/ACM International Symposium on Augmented and Mixed Reality (ISMAR'03)*, pages 198–207. ACM Press, 2003.

[BKM99] Reinhold Behringer, Gudrun Klinker, and David W. Mizell, editors. *Augmented reality: placing artificial objects in real scenes*, 1999.

[Bli77] James F. Blinn. Models of light reflection for computer synthesized pictures. In ACM press, editor, *proceedings of ACM Siggraph '77 (Computer Graphics)*, pages 192–198, 1977.

[Bli88] James F. Blinn. Me and my (fake) shadow. *IEEE Computer Graphics Applications*, 8(1):82–86, 1988.

[BM03] Wolfgang Boehler and Andreas Marbs. Investigating laser scanner accuracy. In *CIPA XIXth international symposium*, 2003.

[BN76] James F. Blinn and Martin E. Newell. Texture and reflection in computer generated images. *Communications of the ACM*, 19(10):542–547, 1976.

[Bri] Brightside Technologies. Dr37-p. www.brightsidetech.com.

[Bro92] Lisa Gottesfeld Brown. A survey of image registration techniques. *ACM Computing Survey*, 24(4):325–376, 1992.

[Cam] Camtronics, Inc. www.camtronics-cnc.co.

[Can] Canon. 10d eos. www.canon.co.uk/for_home/product_finder/cameras/digital_slr/eos_10d/.

[Can86] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[CCWG88] Michael F. Cohen, Shenchang Eric Chen, John R. Wallace, and Donald P. Greenberg. A progressive refinement approach to fast radiosity image generation. In *proceedings of ACM Siggraph '88 (Computer Graphics)*, Annual Conference Series, pages 75–84. ACM Press, 1988.

[Che95] Shenchang Eric Chen. QuickTime VR — an image-based approach to virtual environment navigation. In *proceedings of ACM Siggraph '95 (Computer Graphics)*, volume 29 of *Annual Conference Series*, pages 29–38, 1995.

[CJ02] Christophe Costers and Katrien Jacobs. Ename: Een nieuwe generatie tijdsvensters. Master's thesis, Katholieke Universiteit Leuven, Faculteit Toegepaste Wetenschappen: ESAT, 2002.

[CL89] Patrick Cavanagh and Yvan Leclerc. Shape from shadows. *Experimental Psychology: Human Perception and Performance*, 15:3–27, 1989.

[CON99] Brian Cabral, Marc Olano, and Philip Nemec. Reflection space image based rendering. In *proceedings of ACM Siggraph '99 (Computer Graphics)*, Annual Conference Series, pages 165–170. ACM Press/Addison-Wesley Publishing Co., 1999.

[CR96] Young-Chang Chang and John F. Reid. Rgb calibration for color image analysis in machine vision. *IEEE Transactions on Image Processing*, 5(10):1414–1422, 1996.

[Cro77] Franklin C. Crow. Shadow algorithms for computer graphics. In *proceedings of ACM Siggraph '77 (Computer Graphics)*, Annual Conference Series, pages 242–248. ACM Press, 1977.

[CT81] Robert L. Cook and Kennneth E. Torrance. A reflectance model for computer graphics. In *proceedings of ACM Siggraph '81 (Computer Graphics)*, volume 15, pages 307–316, 1981.

[Dav03] Andrew Davison. Real-time simultaneous localization and mapping with a single camera. In *Proceedings of the IEEE Internation Conference on Computer Vision (ICCV) '03*, 2003.

[DBY98] Paul Debevec, George Borshukov, and Yizhou Yu. Efficient view-dependent image-based rendering with projective texture-mapping. In Springer-Verlag, editor, *proceedings of the 9th Eurographics Workshop on Rendering (Rendering Techniques '98)*, pages 105–116, June 1998.

[DCWP02] Kate Devlin, Alan Chalmers, Alexander Wilkie, and Werner Purgathofer. Tone reproduction and physically based spectral rendering. In *State of the Art Report at Eurographics '02*, 2002.

[Deb98] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *proceedings of ACM Siggraph '98 (Computer Graphics)*, volume 32 of *Annual Conference Series*, pages 189–198, 1998.

[Deb04] Paul Debevec et al. Estimating surface reflectance properties of a complex scene under captured natural illumination. Technical report, USC ICT Technical Report ICT-TR-06.2004, December 2004.

[DHT+00] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In *proceedings of ACM Siggraph '00 (Computer Graphics)*, pages 145–156. ACM Press/Addison-Wesley Publishing Co., 2000.

[DK02] Stanislav Durala and Richard Kittler. CIE general sky standard defined luminance distributions. In *proceedings of eSim 2002*, Canada, 2002.

[DM97] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *proceedings of ACM Siggraph '97 (Computer Graphics)*, volume 31 of *Annual Conference Series*, pages 369–378, 1997.

[DRB97] George Drettakis, Luc Robert, and Sylvain Bugnoux. Interactive common illumination for computer augmented reality. In *proceedings of the 8th Eurographics Workshop on Rendering (Rendering Techniques '97)*, Saint Etienne, France, June 1997.

[DS97] George Drettakis and François X. Sillion. Interactive update of global illumination using a line-space hierarchy. In *proceedings of ACM Siggraph '97 (Computer Graphics)*, volume 31 of *Annual Conference Series*, pages 57–64, August 1997.

[DTM96] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *proceedings of ACM Siggraph '96 (Computer Graphics)*, volume 30 of *Annual Conference Series*, pages 11–20, 1996.

[Elk65] Jack M. Elkin. A deceptively easy problem. *Mathematics Teacher*, 58:194–199, 1965.

[Ena] Ename. Tijdsvensters. www.ename974.org.

[Eos] Eos Systems Inc. Photomodeller. www.photomodeler.com.

[Fau92] Olivier Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig. In *Proceedings of the 2nd European Conference on Computer Vision*, volume 588, pages 563–578, May 1992.

[Fau93] Oliver Faugeras. *Three-dimensional computer vision - A geometric viewpoint.* MIT press, 1993.

[FGR93] Alain Fournier, Atjeng S. Gunawan, and Chris Romanzin. Common illumination between real and computer generated scenes. In *proceedings of Graphics Interface '93*, pages 254–262, Toronto, Canada, May 1993.

[FMS93] Steven Feiner, Blair Macintyre, and Doree Seligmann. Knowledge-based augmented reality. *Communications of the ACM*, 36(7):53–62, 1993.

[FRL+98] Olivier Faugeras, Luc Robert, Stephane Laveau, Gabriella Csurka, Cyril Zeller, Cyrille Gauclin, and Imad Zoghlami. 3-D reconstruction of urban scenes from image sequences. *CVGIP : Image Understanding*, 69(3):292–309, 1998.

[GBR+01] Guy Godin, J.-Angelo Beraldin, Marc Rioux, Marc Levoy, Luc Cournoyer, and Francois Blais. An assessment of laser range measurement of marble surfaces. In *Fifth Conference on optical 3-D measurement techniques*, 2001.

[GCHH03] Simon Gibson, Jon Cook, Toby Howard, and Roger Hubbold. Rapid shadow generation in real-world lighting environments. In *proceedings of the 13th Eurographics workshop on Rendering (Rendering Techniques '03)*, pages 219–229. Eurographics Association, 2003.

[GHH01] Simon Gibson, Toby Howard, and Roger Hubbold. Flexible image-based photometric reconstruction using virtual light sources. In *proceedings of Eurographics 2001*, September 2001. Manchester, UK.

[GM00] Simon Gibson and Alan Murta. Interactive rendering with real-world illumination. In *proceedings of 11th Eurographics Workshop on Rendering (Rendering Techniques '00)*, June 2000. Springer Wien.

[GN03a] Michael D. Grossberg and Shree K. Nayar. High dynamic range from multiple images: Which exposures to combine? In *proceedings of ICCV Workshop on Color and Photometric Methods in Computer Vision (CPMCV)*, Nice, France, October 2003.

[GN03b] Michael D. Grossberg and Shree K. Nayar. What is the space of camera response functions? *Computer Vision and Pattern Recognition (CVPR)*, 02:602, 2003.

[Gre86] Ned Greene. Environment mapping and other applications of world projections. *IEEE Computer Graphics Applications*, 6(11):21–29, 1986.

[Har97] Richard Hartley. In defence of the eight point algorithm. *IEEE International Conference on Computer Vision*, 19(6):580–593, June 1997.

[HD90] Ferdinand Hurter and Vero C. Driffield. Photochemical investigations and a new method of determination of the sensitiveness of photographic plates. *society of chemical industry*, 9:455469, 1890.

[HDH03] Michael Haller, Stephen Drab, and Werner Hartmann. A real-time shadow approach for an augmented reality application using shadow volumes. In *proceedings of the ACM symposium on Virtual reality software and technology (VRST '03)*, 2003.

[HDR] HDRshop. www.HDRshop.com.

[Hei91] Tim Heidmann. Real shadows, real time. *Iris Universe*, 18:23–31, 1991.

[HGC92] Richard Hartley, Rajiv Gupta, and Tom Chang. Stereo from uncalibrated cameras. In *proceedings of Computer Vision and Pattern Recognition*, pages 761–764. IEEE Computer Society Press, 1992.

[HGM00] Toby Howard, Simon Gibson, and Alan Murta. Virtual environments for scene of crime reconstruction and analysis. In *proceedings of SPIE Electronic Imaging 2000*, volume 3960, Jan 2000.

[HH73] Ralph Norman Haber and Maurice Eugene Hershenson. *The psychology of visual perception*. New York: Holt, Rinehart and Winston., 1973.

[HHP+92] Xiao D. He, Patrick O. Heynen, Richard L. Phillips, Kenneth E. Torrance, David H. Salesin, and Donald P. Greenberg. A fast and accurate light reflection model. In *proceedings of ACM Siggraph '92 (Computer Graphics)*, volume 26, pages 253–254, 1992.

[HLHS03] Jean-Marc Hasenfratz, Marc Lapierre, Nicolas Holzschuch, and François Sillion. A survey of real-time soft shadows algorithms. In *proceedings of Eurographics '03*, 2003. State-of-the-Art Report.

[HS99] Wolfgang Heidrich and Hans-Peter Seidel. Realistic, hardware-accelerated shading and lighting. In *proceedings of ACM Siggraph '99 (Computer Graphics)*, pages 171–178. ACM Press/Addison-Wesley Publishing Co., 1999.

[HSA91] Pat Hanrahan, David Salzman, and Larry Aupperle. A rapid hierarchical radiosity algorithm. In *proceedings of ACM Siggraph '91 (Computer Graphics)*, volume 25 of *Annual Conference Series*, pages 197–206, 1991.

[HTSG91] Xiao D. He, Kenneth E. Torrance, Francois X. Sillion, and Donald P. Greenberg. A comprehensive physical model for light reflection. In *proceedings of ACM Siggraph '91 (Computer Graphics)*, pages 175–186, New York, NY, USA, 1991. ACM Press.

[Inta] Integra. Renoir. www.integra.co.jp/eng/products/renoir.

[Intb] International Standard ISO 15739-2003. Dynamic range of an image.

[JAL+04] Katrien Jacobs, Cameron Angus, Céline Loscos, Jean-Daniel Nahmias, Alex Reche, and Anthony Steed. Automatic consistent shadow generation for augmented reality: a demo, 2004. www.cs.ucl.ac.uk/staff/k.jacobs/research/researchcommonillumination.html.

[JAL+05] Katrien Jacobs, Cameron Angus, Céline Loscos, Jean-Daniel Nahmias, Alex Reche, and Anthony Steed. Automatic consistent shadow generation for augmented reality. In *proceedings of Graphics Interface '05*, 2005.

[JL04] Katrien Jacobs and Céline Loscos. Classification of illumination methods for mixed reality. In *State of the Art Report at Eurographics '04*, 2004.

[JL06] Katrien Jacobs and Céline Loscos. Classification of illumination methods for mixed reality. *Computer Graphics Forum*, 26(1):29–51, 2006.

[JNP+95] Pierre Jancene, Fabrice Neyret, Xavier Provot, Jean-Philippe Tarel, Jean-Marc Vezien, Christophe Meilhac, and Anne Verroust. Computing the interactions between real and virtual objects in video sequences. In *proceedings of 2nd IEEE Workshop on Networked Realities*, pages 27–40, October 1995.

[JNVL06a] Katrien Jacobs, Anders H. Nielsen, Jeppe Vesterbaek, and Céline Loscos. Coherent radiance capture of scenes under changing illumination conditions for relighting applications. Technical Report RN/06/20, University College London, 2006.

[JNVL06b] Katrien Jacobs, Anders H. Nielsen, Jeppe Vesterbaek, and Céline Loscos. Reflect: relighting for dynamically lit scenes: a demo, 2006. www.cs.ucl.ac.uk/staff/k.jacobs/research/researchrelighting.html.

[Jol97] Ferenc Jolesz. Image-guided procedures and the operating room of the future. *Radiology*, 204:601–612, 1997.

[JSR04] Guo Jing, Chng Eng Siong, and Deepu Rajan. Foreground motion detection by difference-baed spatial temporal entropy image. In *IEEE Tencon*, 2004.

[JWL05] Katrien Jacobs, Greg Ward, and Céline Loscos. Automatic high dynamic range image generation for dynamic scenes, July 2005. Technical Sketch at Siggraph 2006.

[Kaj86] James T. Kajiya. The rendering equation. In *proceedings of ACM Siggraph '86 (Computer Graphics)*, pages 143–150, New York, NY, USA, 1986. ACM Press.

[KBBM99] Hirokazu Kato, Mark Billinghurst, Rob Blanding, and Richard May. Artoolkit. Technical report, Hiroshima City University, 1999.

[KM00] Jan Kautz and Michael D. McCool. Approximation of glossy reflection with prefiltered environment maps. In ccc, editor, *proceedings of Graphics Interface '00*, pages 119–126, 2000.

[KMH95] Craig Kolb, Don Mitchell, and Pat Hanrahan. A realistic camera model for computer graphics. In *proceedings of ACM Siggraph '95 (Computer Graphics)*, pages 317–324, 1995.

[KUWS03] Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High dynamic range video. In *proceedings of ACM Siggraph '03 (Computer Graphics)*, pages 319–325. ACM Press, 2003.

[LAA+05] Samuli Laine, Timo Aila, Ulf Assarsson, Jaakko Lehtinen, and Tomas Akenine-Möller. Soft shadow volumes for ray tracing. In *proceedings of ACM Siggraph '05 (Computer Graphics)*, volume 24, pages 1156–1165. ACM, 2005.

[Lar92] Greg W. Larson. Real pixels. In James Arvo, editor, *Graphics Gems II*, pages 80–83, 1992.

[LCTS05] Patrick Ledda, Alan Chalmers, Tom Troscianko, and Helge Seetzen. Evaluation of tone mapping operators using a high dynamic range display. In *proceedings of ACM Siggraph '05 (Computer Graphics)*. ACM Press, August 2005.

[LD00] Céline Loscos and George Drettakis. Low-cost photometric calibration for interactive relighting. In *Proceedings of the First French-British International Workshop on Virtual Reality*, Jul 2000.

[LDR00] Céline Loscos, George Drettakis, and Luc Robert. Interactive virtual relighting of real scenes. *IEEE Transactions on Visualization and Computer Graphics*, 6(3):289–305, July-September 2000.

[Lei] Leica. Hds 3000. www.leica-geosystems.com.

[LFD+99] Céline Loscos, Marie-Claude Frasson, George Drettakis, Bruce Walter, Xavier Granier, and Pierre Poulin. Interactive virtual relighting and remodeling of real scenes. In D. Lischinski and G.W. Larson, editors, *proceedings of 10th Eurographics Workshop on Rendering (Rendering Techniques '99)*, volume 10, pages 235–246, New York, NY, Jun 1999. Springer-Verlag/Wien.

[LFTG97] Eric P. F. Lafortune, Sing-Choong Foo, Kenneth E. Torrance, and Donald P. Greenberg. Non-linear approximation of reflectance functions. In *proceedings of ACM Siggraph '97 (Computer Graphics)*, pages 117–126. ACM Press/Addison-Wesley Publishing Co., 1997.

[LH81] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(5828):133–135, 1981.

[LJPP06] Céline Loscos, Katrien Jacobs, Gustavo Patow, and Xavier Pueyo. Inverse rendering: From concept to applications. Tutorial at Eurographics 2006, September 2006.

[LKG+03] Hendrik P. A. Lensch, Jan Kautz, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Image-based reconstruction of spatial appearance and geometric detail. *ACM Transactions on Graphics (TOG)*, 22(2):234–257, 2003.

[LLA06] Jaakko Lehtinen, Samuli Laine, and Timo Aila. An improved physically-based soft shadow volume algorithm. In *proceedings of Eurographics '06*, volume 25. Eurographics Association and Blackwell Publishing Ltd, 2006.

[Los99] Céline Loscos. *Interactive relighting and remodelling of real scenes for augmented reality*. PhD thesis, iMAGIS-GRAVIR/IMAG-INRIA, 1999. PhD thesis.

[LWR+03] Céline Loscos, Hila Ritter Widenfeld, Maria Roussou, Alexander Meyer, Franco Tecchia, George Drettakis, and et al. The create project: mixed reality for design, education, and cultural heritage with a constructivist approach. In *proceedings of IEEE/ACM International Symposium on Augmented and Mixed Reality (ISMAR'03)*, 2003.

[Mar98] Stephen R. Marschner. *Inverse rendering in computer graphics*. PhD thesis, Department of Computer Graphics, Cornell University, Ithaca, NY, 1998. Program of Computer Graphics, Cornell University, Ithaca, NY.

[Max64] James Clerk Maxwell. A dynamical theory of the electromagnetic field. *Royal Society Transactions*, CLV, 1864. Reprinted in R. A. R. Tricker, *The Contributions of Faraday and Maxwell to Electrical Science*, (Pergamon Press, 1966).

[MDA02] Vincent Masselus, Philip Dutre, and Frederik Anrys. The free-form light stage. In *proceedings of 13th Eurographics Workshop on Rendering (Rendering Techniques '02)*, pages 247–256. Eurographics Association, 2002.

[Mel95] J.P. Mellor. Realtime camera calibration for enhanced reality visualization. In *proceedings of Computer Vision, Virtual Reality, and Robotics in Medicine*, pages 471–475, 1995.

[Meta] MetaCreations. Canoma. www.metacreations.com/products/canoma.

[Metb] Metris. www.3dscanners.com.

[MK94] Paul Milgram and Fumio Kishino. A taxonomy of mixed reality visual displays. *IEICE Transactions on Information Systems*, E77-D(12):1321–1329, December 1994.

[MKMS04] Rafal Mantiuk, Grzegorz Krawczyk, Karol Myszkowski, and Hans-Peter Seidel. Perception-motivated high dynamic range video encoding. In *proceedings of ACM Siggraph '04 (Computer Graphics)*, pages 733–741. ACM press, 2004.

[MN99] Tomoo Mitsunaga and Shree K. Nayar. Radiometric self calibration. In *proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 374–380. Fort Collins, June 1999.

[MNP+99] David K. McAllister, Lars Nyland, Voicu Popescu, Anselmo Lastra, and Chris McCue. Real-time rendering of real-world environments. In *proceedings of Eurographics Workshop on Rendering (Rendering Techniques '99)*, pages 145–160, 1999.

[MP95] Steve Mann and Rosalind W Picard. Being undigital with digital cameras: Extending dynamic range by combining differently exposed pictures. In *Proceedings of IST 46th annual conference*, pages 422–428, May 1995.

[MPDW03] Vincent Masselus, Pieter Peers, Philip Dutré, and Yves D. Willems. Relighting with 4d incident light fields. In *proceedings of ACM Siggraph '03 (Computer Graphics)*, volume 22, pages 613–620, New York, NY, USA, 2003. ACM Press.

[MR87] David F McAllister and Woodrow E. Robbins. True three-dimensional imaging techniques and display technologies. In *SPIE Proceedings*, 1987.

[MV91] Allen R. Miller and Emanuel Vegh. Computing the grazing angle of specular reflection. Technical report, Naval Research Lab., Washington, DC, August 1991.

[MW93] Scott MacKenzie and Colin Ware. Lag as a determinant of human performance in interactive systems. In *proceedings of the SIGCHI conference on Human factors in computing systems*, pages 488–493. ACM Press, 1993.

[MYTG94] Vannary Meas-Yedid, Jean-Philippe Tarel, and André Gagalowicz. Calibration métrique faible et construction interactive de modèles 3D de scènes. In *Congrès Reconnaissance des Formes et Intelligence Artificielle*, Paris, France, 1994. AFCET.

[MZ01] Yu-Fei Ma and Hong-Jiang Zhang. Detecting motion object by spatiotemporal entropy, 2001.

[NB03] Shree K. Nayar and Vlad Branzoi. Adaptive dynamic range imaging: Optical control of pixel exposures over space and time. In *proceedings of International Conference on Computer Vision (ICCV)*, 2003.

[NBB04] Shree K. Nayar, Vlad Branzoi, and Terry Boult. Programmable imaging using a digital micromirror array. In *proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[NC96] Ulrich Neumann and Youngkwan Cho. A self-tracking augmented reality system. In *proceedings of the ACM symposium on Virtual reality software and technology (VRST '96)*, pages 109–115, 1996.

[NDM05] Addy Ngan, Frédo Durand, and Wojciech Matusik. Experimental analysis of brdf models. In *proceedings of Eurographics Workshop on Rendering (Rendering Techniques '05)*, pages 117–226. Eurographics Association, 2005.

[Neu98] Peter Neumann. Reflections on reflection in a spherical mirror. *American Mathematical Monthly*, 105:523–528, 1998.

[NHIN86] Eihachiro Nakamae, Koichi Harada, Takao Ishizaki, and Tomoyuki Nishita. A montage method: the overlaying of the computer generated images onto a background photograph. In *proceedings of ACM Siggraph '86 (Computer Graphics)*, pages 207–214. ACM Press, 1986.

[NV06] Anders H. Nielsen and Jeppe H. Vesterbaek. Inverse reflectometry under natural illumination conditions, June 2006. Masters Thesis, University College London and Aalborg University.

[NVi] NVidia. Cube environment mapping. www.nvidia.com/object/feature_cube.html.

[Nyl] Lars S. Nyland. Capturing dense environmental range information with a panning, scanning laser rangefinder. www.cs.unc.edu/~ibr/projects/rangefinder.

[OD00] Takayuki Okatani and Koichiro Deguchi. Estimation of illumination distribution using a specular sphere. In *procceding of the 15th International Conference on Pattern Recognition*, volume 3, 2000.

[OT99] Yuichi Ohta and Hideyuki Tamura. *Mixed Reality - merging real and virtual worlds*. Ohmsha and Springer-Verlag, 1999. Chapter 1, by Paul Milgram and Herman Colquhoun Jr.

[PD03] Pieter Peers and Philip Dutré;. Wavelet environment matting. In *proceedings of Eurographics Workshop on Rendering (Rendering Techniques '03),*, pages 157–166, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[PH] Matt Pharr and Greg Humphreys. Physcical based rendering pbrt. http://pbrt.org.

[Pho] Photomatix. Multimediaphoto. www.hdrsoft.com.

[Pho75] Bui Tuong Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317, 1975.

[PMGT01] Andrea Prati, Ivana Mikic, Costantino Grana, and Mohan M. Trivedi. Shadow detection algorithms for traffic flow analysis: a comparative study. In *IEEE Intelligent Transportation Systems Conference Proceedings*, 2001.

[POF98] Pierre Poulin, Mathieu Ouimet, and Marie Claude Frasson. Interactively modeling with photogrammetry. In *proceedings of Eurographics Workshop on Rendering (Rendering Techniques '98)*, pages 93–104, June 1998.

[PP03] Gustavo Patow and Xavier Pueyo. A survey on inverse rendering problems. *Computer Graphics Forum*, 22(4):663–687, 2003.

[PSC01] Daniel Pletinckx, Neil Silberman, and Dirk Callebaut. Presenting a monument in restoration: the saint laurentius church in ename and its role in the francia media heritage initiative. In *VAST '01: Proceedings of the 2001 conference on Virtual reality, archeology, and cultural heritage*, pages 197–204, New York, NY, USA, 2001. ACM Press.

[PSS99] Arcot J. Preetham, Peter Shirley, and Brian Smits. A practical analytic model for daylight. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 91–100, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[PTYG00] Sumanta N. Pattanaik, Jack Tumblin, Hector Yee, and Donald P. Greenberg. Time-dependent visual adaptation for fast realistic image display. In *proceedings of ACM Siggraph '00 (Computer Graphics)*, pages 47–54, 2000.

[Ras] Rascal. Radiometric self calibration. www1.cs.columbia.edu/CAVE/tomoo/RRHomePage/rrhome.html.

[RBA+95] E. Rose, D. Breen, K Ahlers, D. Greer, C. Crampton, M. Tuceryan, and R. Whitaker. Annotating real-world objects using augmented vision. In *Computer Graphics International '95*, 1995.

[Rea] Realviz. Image modeller. www.realviz.com.

[RFZ05] Anthony Steed Russell Freeman and Bin Zhou. Rapid scene modelling, registration and specification for mixed reality systems. In *ACM Virtual Reality Software and Technology (VRST)*, pages 147–150, 2005.

[RPV93] Holly E. Rushmeier, Charles Patterson, and Aravindan Veerasamy. Geometric simplification for indirect illumination calculations. In *proceedings of Graphics Interface '93*, pages 227–236, May 1993.

[RSSF02] Erik Reinhard, Michael Stark, Peter Shirley, and James Ferwerda. Photographic tone reproduction for digital images. *ACM Transactions on Graphics (TOG)*, 21(3):267–276, 2002.

[RWPD05] Erik Reinhard, Greg Ward, Sumanta Pattanaik, and Paul Debevec. *High Dynamic Range Imaging: Acquisition, Display and Image-Based Lighting*. Morgan Kaufmann Publishers, 2005.

[SAG94] Brian Smits, James Arvo, and Donald Greenberg. A clustering algorithm for radiosity in complex environments. In *proceedings of ACM Siggraph '94 (Computer Graphics)*, volume 28 of *Annual Conference Series*, pages 435–442, 1994.

[Sai95] Kato Saito. Electronic image pickup device, February 1995. Japanese Patent 07-254965.

[SCE04] E. Salvador, A. Cavallaro, and T. Ebrahimi. Cast shadow segmentation using invariant color features. *Computer vision and image understanding*, 95(2):238–259, 2004.

[Sch94] Christophe Schlick. A survey of shading and reflectance models. *Computer Graphics Forum*, 13(2):121–131, 1994.

[SCT+94] Andrei State, David T. Chen, Chris Tector, Andrew Brandt, Hong Chen, Ryutarou Ohbuchi, Mike Bajura, and Henry Fuchs. Case study: Observing a volume rendered fetus within a pregnant patient. Technical Report TR94-034, University of North Carolina, Chapel Hill, 18, 1994.

[SGE01] E. Salvador, P. Green, and T. Ebrahimi. Shadow identification and classification using invariant color models. In *proceedings of ICASSP '01*, volume 3, pages 1545–1548. IEEE, 2001.

[Sha84] Claude E. Shannon. A mathematical theory of communicationr. *Bell System Technical Journal*, 27:379–423 and 623–656, July and October 1984.

[SHC+94] Andrei State, Gentaro Hirota, David T. Chen, William F. Garrett, and Mark A. Livingston. Superior augmented reality registration by integrating landmark tracking and magnetic tracking. In Holly Rushmeier, editor, *proceedings of ACM Siggraph '94 (Computer Graphics)*, volume 30 of *Annual Conference Series*, pages 429–438, 1994.

[SHS+04] Helge Seetzen, Wolfgang Heidrich, Wolfgang Stuerzlinger, Greg Ward, Lorne Whitehead, Matthew Trentacoste, Abhijeet Ghosh, and Andrejs Vorozcovs. High dynamic range display systems. *ACM Transactions on Graphics (TOG*, 23(3):760–768, 2004.

[Sil95] François X. Sillion. A unified hierarchical algorithm for global iillumination with scattering volumes and object clusters. *IEEE Transactions on Visualization and Computer Graphics*, 1(3):240–254, 1995.

[SMa] SMaL Camera Technologies. Smal ultra pocket camera. www.smalcameras.com.

[SP94] François Sillion and Claude Puech. *Radiosity and Global Illumination*. Morgan Kaufmann Publishers, San Francisco, 1994.

[Sph] Spheron. Spheroncam hdr. www.spheron.com.

[SSC02] Mel Slater, Anthony Steed, and Yiorgos Chrysanthou. *Computer graphics and virtual environments: from realism to real-time*. Addison-Wesley Longman Publishing Co., Inc., 2002. Chapter 3: Lighting, the radiance equation.

[SSI99] Imari Sato, Yoichi Sato, and Katsushi Ikeuchi. Acquiring a radiance distribution to superimpose virtual objects onto a real scene. *IEEE Transactions on Visualization and Computer Graphics*, 5(1):1–12, 1999.

[ST04a] Peter Sand and Seth Teller. Video matching. In *proceedings of ACM Siggraph '04 (Computer Graphics)*, volume 23, pages 592–599, New York, NY, USA, 2004. ACM Press.

[ST04b] Peter Sand and Seth Teller. Video matching. Technical report, MIT, 2004.

[SWI97] Yoichi Sato, Mark D. Wheeler, and Katsushi Ikeuchi. Object shape and reflectance modeling from observation. In *proceedings of ACM Siggraph '97 (Computer Graphics)*, volume 31 of *Annual Conference Series*, pages 379–388, 1997.

[TA05] Alejandro Troccoli and Peter K. Allen. Relighting acquired models of outdoor scenes. In *3DIM*, 2005.

[TI04] Akihiko Torii and Atsushi Imiya. Panoramic image transform of omnidirectional images using discrete geometry techniques. *Second International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT'04)*, 00:608–615, 2004.

[TR93] Jack Tumblin and Holly Rushmeier. Tone reproduction for realistic images. *Computer Graphics and Applications*, pages 42–48, 1993.

[TU00] Philippe Thevenaz and Michael Unser. Optimization of mutual information for multiresolution image registration. *IEEE Transactions on Image Processing*, 9:2083–2099, 2000.

[VF94] Douglas Voorhies and Jim Foran. Reflection vector shading hardware. In *proceedings of ACM Siggraph '94 (Computer Graphics)*, pages 163–166. ACM Press, 1994.

[Vio95] Paul Viola. *Alignment by maximization of mutual information.* PhD thesis, Massachusetts Institute of Technology Artificial Intelligence Laboratory, 1995.

[VW95] Paul Viola and William M. Wells. Alignment by maximization of mutual information. In *proceedings of the 5th International Conference on Computer Vision (ICCV '95)*, 1995.

[Wal92] Jorg Waldvogel. The problem of the circular billiard. *Elemente der Mathematik*, 47:108–113, 1992.

[Wal05] Bruce Walter. Notes on the Ward BRDF. Technical report, Cornell Program of Computer Graphics, 2005.

[Wan95] B. A. Wandell. *Foundations of Vision.* Sinauer Associates, Inc., 1995.

[War92] Gregory J. Ward. Measuring and modeling anisotropic reflection. In *proceedings of ACM Siggraph '92 (Computer Graphics)*, pages 265–272. ACM Press, 1992.

[War94] Gregory J. Ward. The radiance lighting simulation and rendering system. In *proceedings of ACM Siggraph '94 (Computer Graphics)*, pages 459–472. ACM Press, 1994.

[War04] Greg Ward. Fast, robust image registration for compositing high dynamic range photographs from handheld exposures. *Graphics Tools*, 8(2):17–30, 2004.

[Web] Web 3D consortium. www.web3d.org.

[WGT+05] Andreas Wenger, Andrew Gardner, Chris Tchou, Jonas Unger, Tim Hawkins, and Paul Debevec. Performance relighting and reflectance transformation with time-multiplexed illumination. In *proceedings of ACM Siggraph '05 (Computer Graphics)*, volume 24, pages 756–764, New York, NY, USA, 2005. ACM Press.

[WRP97] Greg Ward, Holly Rushmeier, and Christine PiatKo. A visibility matching tone reproduction operator for high dynamic range scenes. *IEEE Transactions on Visualization and Computer Graphics*, 3:291306, 1997.

[YDMH99] Yizhou Yu, Paul Debevec, Jitendra Malik, and Tim Hawkins. Inverse global illumination: recovering reflectance models of real scenes from photographs. In *proceedings of ACM Siggraph '99 (Computer Graphics)*, pages 215–224. ACM Press/Addison-Wesley Publishing Co., 1999.

[YM98] Yizhou Yu and Jitendra Malik. Recovering photometric properties of architectural scenes from photographs. In *proceedings of ACM Siggraph '98 (Computer Graphics)*, pages 207–217. ACM Press, 1998.

[Zha00] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

Appendix A

Glossary

Definitions and abbreviations used:

AR	See *Augmented Reality*.
Augmented Reality	A reality that consist of both real and virtual objects, but the majority of the objects is real. In other words: the real environment is augmented with virtual objects. Augmented realities form a subgroup within the mixed realities; often the term augmented reality is preferred to the more general expression of mixed reality if the application runs in real-time. See also *Mixed Reality*.
Augmented Virtuality	A reality that consist of both real and virtual objects, but the majority of the objects is virtual. In other words: the virtual environment is augmented with real objects. Augmented virtualities form a subgroup of mixed reality. See also *Mixed Reality*
AV	See *Augmented Virtuality*.
BRDF	Bi-directional reflectance distribution function. Gives the ratio of incoming irradiance from a certain direction with the radiance leaving into another direction for a point p of a certain material. Often written as $f(p, \omega_i, \omega_r)$, with ω_i the incoming direction and ω_r, the outgoing direction. The BRDF is the mathematical interpretation of a reflectance.
CAR	See *Computer Augmented Reality*.
Coherent radiance distribution	A scene radiance distribution is coherent when the radiance of all scene points are solutions of radiance equations containing the same initial light sources, and the same geometric constitution. See also *Radiance distribution*.
Common illumination	A virtual illumination method for mixed reality that calibrates the lighting effects across the virtual and real objects without actually changing the illumination conditions present in the real scene. Typically the change in lighting effects induced by the virtual objects are simulated, such as shadows cast by the virtual objects onto the real environment, or real shadows that are cast onto the virtual objects. See also *Illumination method*, *Lighting effects*, *Mixed Reality*.

Computer Augmented Reality	A type of AR that visualizes the real and virtual elements using a non-immersive display (like a computer screen), rather than using a see-through device. Usually CAR operates from a pre-defined 3D model of the scene. See also *AR*, *Augmented Reality*.
Consistent	Used in the context of virtual shadow generation, but it can be used for other types of lighting effects too. It means that the virtual shadows are conform to the real shadows: cast in the same direction and with the same appearance such as the colour. See also *Lighting effects*.
Distortion	The error in a latitude-longitude image generated from a lightprobe image, after assuming that the 3D scene is infinitely far from the reflective sphere used to capture the lightprobe image. See also *Latitude-longitude image*, *Lightprobe image*.
Exposure	This concept can have two different meanings. The first is the exposure setting of a camera, which is the product of the camera's shutter speed times its aperture width. The second meaning is an image captured as part of a sequence of LDRIs with different exposure settings. See also *LDRI*.
Geometric model	Also called geometry, reconstruction or reconstructed geometry: the 3D geometry of a (real or virtual) scene. The model can be obtained with a scanner or reconstruction software (for a real scene) or with modelling software (for a virtual scene). Usually the geometric model is presented as a mathematical description of the scene surface points or as a triangular mesh.
Global illumination	Usually used in the context of rendering a certain type of illumination in a scene (real, virtual, or MR). Global illumination calculates the illumination in a scene while taking into account the direct and indirect light sources. As a result (soft) shadows and inter-reflections are simulated. See also *Local illumination*.
HCM	See *High Contrast Movement*.
HDR	High Dynamic Range.
HDRI	See *High Dynamic Range Image*.
High Contrast Movement	Type of movement for which there is a high contrast between the dynamic object(s) and the background. Examples are for instance cars and people moving in a street, planes in the sky, etc.
High Dynamic Range Image	A floating point image representing the camera sensor's irradiance without saturation. Usually obtained by combining several LDRIs captured with changing exposure settings. See also *LDRI*, *Exposure*.
Identity map	An image associated with a latitude-longitude image LL, and having the same dimensions. The colour channels define uniquely the identity number of the scene triangle that the corresponding pixel in LL belongs

	to. The identity map is equal to the first two channels of the position map. See also, *Latitude-longitude image*, *Position map*.
Identity texture	A 4-channel floating point texture, associated with an illumination cluster. Each pixel in the identity texture defines its position in the illumination cluster, and thus in the 3D scene. The identity texture is used to create the position map. See also *Illumination Cluster*, *Position map*.
Illumination	Usually used in the context of scene illumination or the illumination of an object. It refers to the radiance that is present in the scene. The illumination of an object can be split in direct light and indirect light. In a way it is similar to the radiance distribution of a scene. See also *Radiance distribution*.
Illumination Cluster	A group of triangles part of a 3D scene and part of a certain material cluster. For all triangles part of the same illumination cluster, the texture is extracted from the same input image I. See also *Input image*, *Material Cluster*.
Illumination method	In the context of this thesis, an illumination method is a method used to generate the lighting effects in a mixed reality. See also *Lighting effect*, *Mixed Reality*.
Illumination pattern	The set-up of the light sources in the scene.
Illumination setting	The set-up of the light sources in the scene.
Input image	Depending on the context in which it is used, it can have two different meanings. When referred to in the context of an HDRI, an input image is an LDRI part of the exposure sequence captured to generate the HDRI. In the context of inverse illumination it is an image of a certain part of a 3D scene, used to model the scene and used to estimate the BRDF properties of the surfaces visible in the input image. See also *BRDF*, *Exposure*, *HDRI*, I_i.
Inverse illumination	The process used to estimate the reflectance properties of the materials in the real scene by applying the inverse of the radiance equation. Inverse illumination can then be used to obtain a common illumination or relighting of the mixed reality. When applied to relighting it is referred to as physically-based illumination. See also *Common illumination*, *Radiance equation*, *Relighting*.
Latitude-longitude image	An image of a lightprobe transferred to latitude-longitude space. The original image, when cropped to the dimensions of the sphere, can be mapped onto a sphere. The latitude-longitude image represents this mapping, where the vertical axis represents the latitude and the horizontal axis represents the longitude coordinates. Each pixel can now be mapped to its latitude-longitude coordinate on a sphere. See also *Lightprobe*.
LCM	See *Low Contrast Movement*.
LDR	Low Dynamic Range.

LDRI	See *Low Dynamic Range Image.*
Lighting effect	A lighting effect is an effect created by light/object/material interactions, such as a shadow, or glossy highlight. The characteristics of a lighting effect depend on the position and strength of the light sources in the scene, the geometry and the material properties of the objects in the scene.
Lightprobe	Can have two different interpretations depending on the context in which it is used. Either it refers to a reflective sphere, or to an image of a reflective sphere. See also *Lightprobe image.*
Lightprobe image	An image of a reflective sphere, usually used to capture the illumination in the scene.
Lightprobe shadow	The cone-like volume behind the reflective sphere that cannot be seen by the camera as it is obscured by the sphere. See also *Lightprobe.*
Local illumination	Usually used in the context of rendering a certain type of illumination in a scene (real, virtual, or MR). Local illumination calculates the illumination in a scene while taking into account the direct light sources only. As a result usually only (hard) shadows are simulated. See also *Global illumination.*
Low Contrast Movement	Type of movement for which the moving object(s) and background are very similar. Examples are for instance the movement of branches in a tree due to the wind, the rippling of the water, etc.
Low Dynamic Range Image	An integer image representing the scene intensity by integers between 0 and 255. The relation between pixel intensity and scene radiance is defined by the camera's response curve.
Material Cluster	A group of triangles part of a real 3D scene, for which the material properties are the same.
MC	See *Material Cluster.*
Median Threshold Bitmap	A type of transformation which transforms an image into a binary image by applying the median grey value as a threshold to the image's intensity values.
Mixed Reality	An environment comprising both real and virtual objects. The lighting effects between virtual and real objects can be generated using either common illumination, relighting or physically-based illumination. A mixed reality is either an augmented reality or an augmented virtuality. See also *Augmented reality, Augmented virtuality, Illumination method, Lighting effects.*
Movement cluster	Usually used in the context of HDRI generation. It is a group of pixels that are affected by movement in the sequence of exposures used to generate an HDRI. The cluster of pixels should form one well-defined closed group of pixels. See also *Exposures, HDRI.*
MR	See *Mixed Reality.*

MTB	See *Median Threshold Bitmap.*
Neutral Cluster	Is the name for the group of triangles that belong to the geometric model of a 3D scene, but for which no BRDF estimate is made during the inverse illumination calculations. See also *Inverse illumination.*
Patterned	This is a feature assigned to a material cluster when the radiance texture that is projected onto the clusters shows patterns or underlying scene features. Usually this indicates that not all points inside the material cluster share the same BRDF model. See also *BRDF, Material Cluster.*
Physically-based illumination	An illumination method for mixed reality generating either common illumination or relighting using the physical laws of material and light interaction. See *Common illumination, Illumination method, Inverse illumination, Mixed Reality.*
Pinching effect	Visible in a latitude-longitude image of lightprobe image as a result of the assumption that the lightprobe is an infinitesimally small sphere. See also *Latitude-longitude image, Lightprobe.*
Position map	An image associated with a latitude-longitude image LL and having the same dimensions. For each pixel t in LL, the corresponding pixel in the position map defines the position of the scene point in the 3D scene projected onto pixel t. The position map consists of four 32-bit channels. The R- and G-channel define the triangle, and hence the illumination cluster, the pixel belongs to; the B- and α-channel define the position of that pixel within the illumination cluster. See also *Illumination Cluster, Latitude-longitude image.*
Radiance distribution	The radiance distribution of a scene is the collection of radiance that is available for all scene points. It refers to the radiance that is distributed into the scene after enabling the light sources and propagating the light according to the radiance equation.
Radiance equation	A function defining the radiance of a scene point based on the scene's geometry, the radiance of all other scene points, and the reflectance function of the material on which the scene point lies.
Real object(s)	Tangible object(s) part of the user's physical environment or projected onto this latter environment after capturing this/these object(s) using a camera device.
Real scene (environment)	Used to refer to those parts of an environment that are tangible and are either part of the user's environment or are displayed onto this latter environment after capturing the tangible environment with a camera device.
Real shadow	The shadow of a real object cast onto the real environment.
Real shadow estimate	The estimated position of the real shadow, based on the geometric model of the 3D scene and the position of the light sources.

Reality Virtuality Continuum	This continuum, presented by Milgram et al. [MK94, OT99], covers all possible environments that can be represented. This continuum consists of four sub-groups: Reality, Augmented Reality, Augmented Virtuality and Virtuality. See also *Augmented reality*, *Augmented virtuality*, *Real scene* and *Virtual scene*.
Reflectometry	The art of measuring (or estimating) the reflectance of an object.
Relighting	A virtual illumination method for mixed reality that calibrates the illumination across the virtual and real objects while actually changing the illumination conditions present previously in the real scene. See also *Illumination method*, *Mixed Reality*.
RV	See *Reality Virtuality Continuum*.
Semi-soft shadow	A shadow usually consists of umbra and penumbra areas. Points that are totally blocked from the (direct) light sources lie in umbra; points that are partially blocked from the (direct) light sources lie in penumbra. A semi-soft shadow consists mainly of umbra, though some penumbra can be visible on the border of the shadow.
Semi-soft shadow map	A semi-soft shadow map differs from a shadow map in that it contains grey-level values instead of being binary. See also *shadow map*.
Shadow factor	A scaling factor that defines the relation between an area in shadow and an area not in shadow. The ambient lighting of the two areas is the same, and the two areas share the same material properties. The scaling factor is defined through a triplet, representing a scaling factor for each colour channel.
Shadow map	A binary mask, which can be projected onto the geometric model of a real 3D scene. For each scene point it defines whether the point is in a real shadow (0) or not inside a real shadow (1).
Tonemapping	This is an image operation that transforms the floating point values of an HDRI to an LDRI, such that the resulting LDRI can be visualised on a regular 24 bit display device. The transformation is often based on how the human visual system perceives radiance values, or on the contrast in the image. The transformation is often non-linear. See also *HDRI*, *LDRI*.
UI	See *Uncertainty Image*.
Uncertainty Image	An uncertainty image defines the uncertainty about a pixel's static status in a sequence of exposures. If the uncertainty is large, the pixel is most likely affected by movement. See also *Exposures*.
Variance Image	A variance image defines the variance of the irradiance values of a pixel over a set of exposures. To calculate the variance image, the camera curve needs to be known in order to transform the intensity values in the exposures to irradiance values. See also *Exposures*.

VI	See *Variance Image.*
Virtual object(s)	Non-tangible object(s) created by a user. Part of the objective of this thesis is to include virtual objects into a real scene while incorporating the lighting effects that this inclusion causes. See also *Lighting effect.*
Virtual scene (environment)	A non-tangible environment, the objects are user-created, usually with computer software.
Virtual shadow	The shadow of a virtual object projected onto either a real or virtual scene. The position of the virtual shadow depends on the scene geometry and the position of the virtual light source.

Mathematical expressions:

(k, l)	The coordinates of a pixel in an image.
Ω	Represents the entire 3D world in terms of solid angle.
$\overline{H}$	A High Dynamic Range Image after movement removal. See also, *High Dynamic Range Image.*
$\overrightarrow{n}$	The normal for a certain scene point.
ρ	The BRDF of a material. See also *BRDF.*
ρ^n	The estimate of the BRDF of a material, at iteration n. See also *BRDF.*
θ	The elevation angle part of the spherical coordinates.
C	The centre of the camera.
D	Distance between the centre of the camera to the centre of the lightprobe.
$d\omega$	An infinitely small solid angle, due to its infinitesimal character it often represents a direction.
E, E_i	Depending on the context in which it is used, E stands for irradiance, or irradiance image. The index i in particularly means the irradiance image resulting from the transformation of input image i to irradiance using the camera curve. See also I_i, *Input Image.*
$f(p, \omega_i, \omega_r)$	The BRDF of a point p, defines the percentage of radiance coming from ω_i that is reflected into direction ω_r. See also *BRDF.*
$H(I)$	The entropy of an image I.
$H(I, J)$	The joint entropy of image I and J.
H_i	Local entropy image for an image I_i. See also *Input image.*
I_i	Input image i. See also *Input image.*
IC_i	Illumination cluster i. See also *Illumination Cluster.*
IM_i	Identity Map i. See also *Identity map.*

IT_i	Identity Texture i. See also *Identity texture*.
$L(p, \omega)$	The reflected radiance of a scene point p, in direction ω.
$L_e(p, \omega)$	The emitted radiance of a scene point p, in direction ω.
LL_i	Latitude-longitude map number i. See also *Latitude-longitude image*.
LP_i	Lightprobe image number i. See also *Lightprobe image*.
MC_i	Material cluster i. See also *Material Cluster*.
MI	Mutual information.
p (p_i)	A scene point. In the world coordinate system the coordinates of p (p_i) is given by (x, y, z) (or (x_i, y_i, z_i)). In the local frame of reference $X'Y'Z'$, the spherical coordinates of p (p_i) are given by: $(\varphi_p, \theta_p, r_p)$ (or $(\varphi_{p_i}, \theta_{p_i}, r_{p_i})$)
P_i	Position map i. See also *Position map*.
q (q_i)	The projection of scene point p (p_i) onto the sphere. The projection is defined by the intersection of the line Sp (or Sp_i) and the sphere. See also figure 6.6.
R_i	The rendering of an image used in the context of inverse illumination. The image R_i is rendered from the same viewpoint as input image I_i, with all materials set to the current BRDF estimate, and the light sources defined by the back-projected textures of LP_i onto the neutral cluster. See also *BRDF*, *Input image*, *Neutral Cluster*.
S	The centre of the reflective sphere.
t, s	A pixel in an image.
T_{UI}	A threshold used to transform UI into UI_T. See also *UI*, UI_T.
T_{VI}	A threshold used to transform VI into VI_T. See also *VI*, VI_T.
UI_T	Uncertainty image after having applied a threshold to detect movement pixels. See also *Uncertainty Image*.
VI_T	Variance image after having applied a threshold to detect movement pixels. See also *Variance Image*.
$W_i(k, l)$	The weight for pixel (k,l) in exposure i. See also *Exposure*.
$X'Y'Z'$	The local coordinate system positioned in the centre S of the sphere.
XYZ	The world coordinate system.

Appendix B

High dynamic rang image formats and encoding standards

The following table list standard HDRI formats and some of their characteristics [RWPD05]:

Format	Encoding type	Compression	Support/Licence
HDR	RGBE	run-length	Open Source
	XYZE	run-length	e.g., Radiance
TIFF	IEEE RGB	none	Public domain
	LogLuv24	none	(libtiff)
	LogLuv32	run-length	
EXR	Half RGB	wavelet, ZIP	Open Source e.g., OpenEXR

Table B.1: *An overview of three different HDRI formats.*

The following table gives a short overview of some standard HDRI encoding formats [RWPD05]:

Encoding	Colour space	Bits / Channel	Dynamic range (log_{10})	Relative error
sRGB	RGB in [0,1]	24	1.6 orders	variable
RGBE	positive RGB	32	76 orders	1.0%
XYZE	(CIE) XYZ	32	76 orders	1.0%
IEEE RGB	RGB	96	79 orders	0.000003%
LogLuv24	Y + (u',v')	24	4.8 orders	1.1%
LogLuv32	Y + (u',v')	32	38 orders	0.3%
Half RGB	RGB	48	10.7 orders	0.1%
scRGB48	RGB	48	3.5 orders	Variable
scRGB-nl	RGB	36	3.2 orders	Variable
scYCC-nl	YCbCr	36	3.2 orders	Variable

Table B.2: *An overview of some popular HDRI encoding formats.*

Appendix C

How to capture a good HDRI

When capturing a high dynamic range image, the photographer should take a few things into consideration. Before discussing several of these issues, some terminology, as introduced in chapters 2 and 5, is repeated here. A high dynamic range image (*HDRI*) is either captured with an HDR camera, or generated from a set of conventional or low dynamic range images (*LDRIs*) captured with different exposure settings (therefore also often called *exposures*). It is the latter approach that was investigated in this thesis, and it is therefore this approach that is discussed in this appendix.

A Static scene and camera

When combining several exposures into an HDRI, it is essential that the exposures contain the same scene content. The scene content comprises the scene geometry and the scene illumination. Ideally the exposures are captured from exactly the same viewpoint, however if the method from chapter 5 is adopted to generate the HDRI, small camera misalignments are allowed. Ideally, the scene contains no moving objects, though most movement can be removed with the method presented in chapter 5. Nevertheless, the movement should never cover 100% of the viewing window in order not to compromise the image alignment. The larger the area in the exposures covered by movement, the less likely the camera alignment will be successful. Also, the larger the area extended by the moving object, the less reliable the HDR content in the final HDRI, as the moving object is represented by only one LDR exposure.

B White balance

When capturing the exposures, the camera's white balance should be disabled. If the specific camera used cannot disable the white balance, it should at least be set to the same, fixed colour temperature. *Automatic white balancing* should never be used. Setting the white balance is equal to setting the colour temperature to that of the main light source in the scene. A white object lit by a specific colour temperature has white image values[1] if the white balance is set to that specific colour temperature.

White balancing an image involves shifting the image values within a colour channel to ensure that an object with the same colour temperature as the white balance receives white pixel values. The amount shifted usually differs between the colour channels (the colour temperature is rarely white, usually more yellowish or reddish). When white balancing is enabled and the shifting is consistent between the different exposures, it is still possible to generate an HDRI, but it is better not to use white balancing at all. It is preferred to capture the LDRIs as RAW images. These can be transported to another format in a pre-processing step, in which the colour balance can be chosen separately.

[1] The R-, G-, and B-channel are identical.

When capturing HDRIs from different viewpoints, it is essential that these images share the same colour temperature. Since the white-balance might change, the pixel values of a scene point in the HDRIs can be inconsistent: the RGB-values of the radiance of the scene point can be shifted along the different HDRIs.

C Auto-focus

If the camera's auto-focus uses a passive system, based on contrast, it should be disabled and the focus of the camera lens needs to be set manually. The reason is that due to the different exposures, the camera sees different features between different exposures used, and therefore most likely applies a different focus setting for the different exposures. This undesired for two reasons. The first is that changing the focus will change the scene content visible in the LDRIs. The second is that changing the focus will change the vignetting effects, between the different LDRIs.

When the user finds it difficult to select a focus setting himself, the auto-focus can be used to find a desirable focus setting for the middle exposure. Before starting the capture of the different exposures, the user selects a manual focus, without changing the focus setting set by the auto-focus.

D Exposure settings

When capturing different HDRIs from different viewpoints for a relighting application, it is advised to use the same exposure settings (range and number) for all HDRIs. This avoids unwanted shifting effects often introduced by HDRI generation software.

The number of exposures used might vary from 3 to 11, depending on the dynamic range of the scene. While a large number of exposures used ensures that the entire dynamic range is captured without saturation, it also gives rise to a higher chance of undesirable object or camera movement.

The user can either change the exposure time, the aperture width, or a combination of both. Usually the accuracy of the *promised* exposure time, is higher than the accuracy of the *promised* aperture width, thought this depends on the manufacturer of the camera. Also, vignetting effects are dependent on the aperture width and changing the aperture therefore increases unwanted differences between the exposures. In this thesis, the exposure speed was changed only.

If various exposures are needed for the same scene but from different viewpoints, the best way to identify the number of exposures to use, is by first capturing the brightest parts in the scene. The exposure time that allows capturing this bright part of the scene without saturation is the lowest exposure time that absolutely needs to be used to capture a scene. Choosing an exposure time one f-stop lower than that is even a better choice. The largest exposure time to use is more difficult to define. Usually the exposure time that captures the darkest part of the scene without turning the image into complete under-exposure should be fine. HDRI generation software usually cancels out those pixels that are saturated, but having an image that is entirely saturated makes no sense.

E Flashlight

Finally, many photographers forget that the illumination should be static across the different exposures. This also includes not using a flash, as the intensity of a flashlight often depends on the exposure setting used. If the HDRI is captured appropriately, the flash is not required anyway, as the radiance contents of the scene visible in the viewing window is available as floating point values. Appropriate tonemapping algorithms can be user afterwards to manipulate (e.g., dodge and burning) the radiance values if desired for aesthetical reasons.

Appendix D

How to capture and model a 3 dimensional HDR model

This appendix provides some hints when capturing and modelling a 3D high dynamic range model using user-aided reconstruction software. We assume that the objective is to capture an HDRI model for relighting, according to the methodology given in chapter 6. Section D.1 discusses how to capture the input images, required for the modelling, and texture extraction. Section D.2 explains how the lightprobe images are captured, and how to use a reflective sphere. Finally section D.3 discusses how to model the scene, including how to export high dynamic range textures.

D.1 Capturing input images

The input images serve three different tasks:

- The input images are used to extract scene features during the 3D modelling,
- The input images are used to extract HDR texture information,
- The input images are used as reference images during the inverse illumination process.

For all three of these tasks, it is necessary that the input images capture clearly the surfaces in the scene. To aid the modelling, it is necessary that there are sufficient scene features visible in each input image. A minimum of three scene features are needed to ensure calibration in ImageModeler [Rea], but usually it is helpful that more scene features are visible. Also, to improve the quality of the reconstruction, it is better to have each scene feature visible in more than two input images. From experience we have learned that if each scene feature appears in at least four different input image, usually a good calibration can be obtained. The scene features ideally do not lie on one plane, but are distributed uniformly across the scene.

Textures are usually extracted per triangle, and this triangle needs to be visible in at least one input image. Therefore it is necessary that when capturing surfaces with an input image, that at least three scene features are visible in the input image that lie on a plane in the 3D scene. Sometimes reconstruction software allows blending of different input images, and extract a blended texture. This would reduce the requirement that each modelled triangle needs to be visible in one input image, as its radiance texture could potentially be created through blending of several input images. However, when reconstructing HDR models, this blending option is undesirable and cannot be used, see also section D.3.2.

To aid the texture extraction and to avoid having to return to the scene to capture additional input images, the photographer should triangulate the scene roughly in his mind, or even better, on paper. When capturing the input images, the photographer should trace back if all triangles are covered in at least one input image.

If the inverse illumination process calculates a diffuse BRDF only, it is better to avoid capturing specular surfaces from the specular angle. This avoids having specular highlights in the texture. Alternatively, when specular BRDF's are extracted, it is usually better to have for each material at least one image that shows a highlight on a surface of that material. The reason is that if no information is available about the specular characteristics of a certain material, it will be impossible to correctly estimate and verify its specular behaviour.

With the method presented in chapter 6, each input image is used to estimate the BRDF values of the surfaces visible in that input image. To obtain this, a lightprobe image is required per input image, positioned near the surfaces visible in the input image. The photographer should take this into account when capturing the input images, by making sure that the surfaces for which the input image are used are also near to each other. Otherwise the lightprobe is near to one surface, but far from the other surfaces. In other words, the input image should focus on surfaces that are near each other, while ensuring that enough scene features are available to guarantee calibration.

D.2 Capturing lightprobe images

For each input image, a lightprobe image needs to be captured. The lightprobe image needs to be positioned near to the surfaces visible in the input image. As was explained in chapter 6, the resolution of the lightprobe image is low. Therefore it is important that the camera zooms in on the lightprobe image, trying to cover the entire viewing window with the specular sphere, to improve the resolution of the lightprobe in the image. Since it is better that the camera is positioned far from the sphere, the use of a tele-lens is desirable. When capturing the lightprobe image, the photographer should take into account that the more central an object is reflected on the sphere, the higher its resolution in the image. Therefore important objects such as light sources should be reflected towards the centre of the sphere, as much as possible, without obstructing the light sources with the camera and tripod.

If possible, the photographer should capture the HDRIs using a timer. This has two advantages. Firstly, the camera will not move due to user intervention. And secondly, the photographer can leave the room, or stand aside in order for its reflection not to cover too many pixels in the final lightprobe image. It should be noted that when a heavy tele-lens is used to capture the lightprobe, small camera movements are nearly always unavoidable, due to the weight of the tele-lens. Image alignment using the method described in chapter 5 is therefore obligatory.

Reflective spheres are easy to obtain. One can use for instance a specular christmas bulb, though such a bulb often contains visible borders where the two halves of the bulb are glued together. An alternative is to use a ballbearing. Ballbearings exist in all sorts of sizes and all sorts of precision. This offers a huge advantage compared to christmas bulbs. The disadvantage is that ballbearings are nearly always covered with an oily substance, which prevents oxidization. It is a real hassle to remove this oily substance, without scratching the ball. Our experience is that *ethanol* is the best solution to remove the grease.

Another problem with using a ballbearing is the positioning of the sphere in the scene. While a christmas bulb is light and can be positioned nearly everywhere, including on a rope dangling from the ceiling, the ballbearings are solid and heavy. For our application, we had a $22mm$ long solid metal cylinder attached to the ball using superglue, see figure D.1 (a). The metal cylinder was rounded on the top, in order to

create a perfect fitting between the sphere and the cylinder. This creates a larger contact area between the sphere and the cylinder and ensures a stronger connection with the glue. Along the length of the tube, a screw hole was drilled, such that the cylinder, and sphere, could be screwed on a tripod, see figure D.1 (b) and (c). This creates flexibility in positioning the sphere at different heights. In the examples shown in chapter 6, the lightprobes were positioned on the floor on top of their support unit.

To prevent scratching the surface of the reflective sphere and adding grease, it is advised to use cotton gloves when handling the lightprobes, as shown in figure D.1 (a) and b), which are sold in nearly every photography shop.

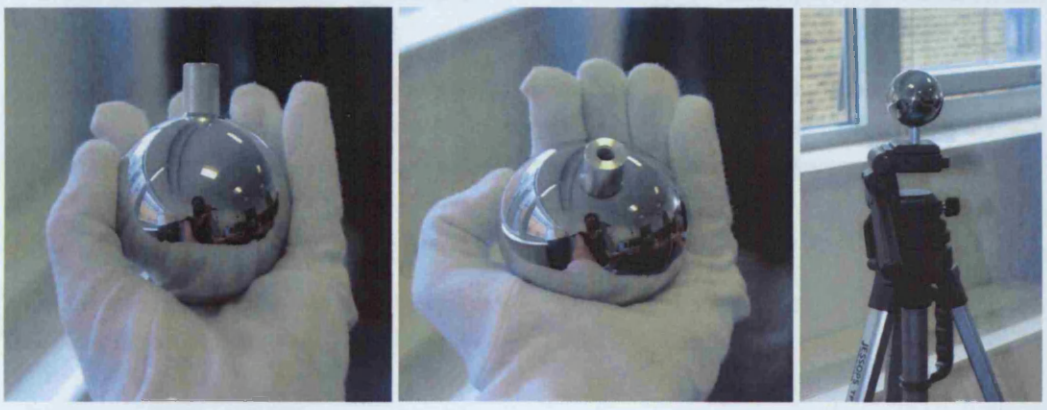


Figure D.1: *Lightprobes: some practical issues. (a) A metal cylinder is attached to the lightprobe, the metal cylinder is rounded at the top to increase the contact area between the cylinder and lightprobe. (b) Inside the metal cylinder a screw hole is drilled. (c) The screw hole enables the positioning of the lightprobe onto a tripod.*

D.3 Modelling using reconstruction software

D.3.1 Creating MCs and ICs

With ImageModeler it is possible to generate different meshes, consisting of a set of triangles. When exporting the scene to VRML, these meshes appear as separate *transform* element in the VRML-file which combines all information of the triangles belonging to a specific mesh (like texture, position, etc.), see also appendix E. For our relighting applications, it is essential to group the triangles of one material into one unique mesh. Such a mesh is then identical to what was defined as a material clusters in chapter 6.

The textures of the triangles should be extracted per group of triangles belonging to the same mesh (or MC) and visible in the same input image. When the input image is used to extract texture for several materials, it is important that the user repeats the texture extraction for each group of triangles belonging to a different material cluster separately. This ensures that in the VRML file, within the *transform* elements different *shape* elements appear. Each such shape is a set of triangles belonging to the mesh defined by the transform and containing textures from only one input image. In other words, such a shape is identical to what was called an illumination cluster in chapter 6.

D.3.2 Creating HDRI textures

ImageModeler does not operate on HDR input images, nor does it extracts HDR textures. As a solution to this problem we adopted the following method:

- Model the scene from the middle exposure of all input images. This exposure should represent the scene the most well-defined. If this is not the case for a certain input image, a different exposure can be chosen for that particular input image.
- Extract the texture for all shapes in the scene, this creates textures objects in ImageModeler.
- After the modelling and the texture extraction, the input images are substituted by the first exposures for each input image. To extract the textures, a *reload* process is applied to the texture objects created in the previous step. Save the textures.
- This is repeated sequentially for all exposures of the input images (each input image should contain the same number of exposures.)
- After this iterative texture extraction process, for each texture of an IC, the same number of LDR textures exist as there are exposures for its associated input image. These LDR textures are now combined into HDR textures using the HDRI generation methodology presented in chapter 5.

D.3.3 Generation of the scene file

When the scene geometry and all the HDR textures are generated, the scene is exported to a VRML file. This file is further translated into an X3D file as is explained in appendix E.

Appendix E

Reflect: a step-by-step manual

E.1 Introduction

In this appendix, the system REFLECT is presented. This system was build for the relighting application presented in chapter 6. We have spent much time on building a user-friendly graphical interface, which takes as input a description of a 3D scene, and allows the user to analyse the scene, calibrate the lightprobes, estimate the BRDF values of the surfaces in the scene, and relight the scene using a novel illumination pattern.

Section E.2 explains the structure of the input file, that needs to be fed into the system. Sections E.3, E.4, E.5, E.6, and E.7, take the reader through the different steps of REFLECT.

Reflect has been build in collaboration with Anders H. Nielsen and Jeppe Vesterbæk.

E.2 Scene description using X3D

In appendix D it was discussed how the HDRI model should be modelled and exported to VRML. This VRML model is then further translated into an X3D file [Web]. X3D is a vector graphics markup language, similar to VRML and written using XML. We used a macro to translate the VRML file into a X3D file. The X3D file requires a few more modifications, extra features, before it can be accepted by REFLECT. These extra features are extra elements and attributes, which are not described by VRML.

The input file contains 4 different sections: *Transforms*, *Shapes*, *Lightprobes*, and *InputImages*. The following pseudo-code presents a typical structure of an input scene:

```
<X3D profile="Full">
  <head></head>
  <Scene>
    <SuggestedView  cx="..." cy="..." cz="..."
                    dx="..." dy="..." dz="..."
                    upx="..." upy="..." upz="..." fov="..." />

    <Transform  DEF="MC1_name"
                translation="tx ty tz"
                scale="sx sy sz"
                rotation="rx ry rz a">
        <Shape >
```

```
        <InputImage USE="Input1" />
        <Appearance >
            <Material  transparency="0.0"/>
            <ImageTexture  url='"textures/texturename.exr"'/>
        </Appearance>
        <IndexedFaceSet
            DEF="IC name"
            coordIndex="c11 c12 c13 -1 c21 c22 c23 -1 ..."
            texCoordIndex="t11 t12 t13 -1 t21 t22 t23 -1 ...">
            <TextureCoordinate DEF="UVs_IC_Object"
                point="p1 p2 p3 p4 p5 p6 ..."/>
            <Coordinate DEF="Verticess_IC_Object"
                point="p1 p2 p3 p4 p5 p6 ..."/>
        </IndexedFaceSet>
    </Shape>
    ...
    <insert as many shapes as necessary>
    ...
</Transform>

....
<insert as many transforms as necessary>
....

<Lightprobes>
    <Lightprobe DEF="LP1_name" radius="...">
        <texture url="'lightprobes/LP1_LL.exr'" />
        <angulartexture url="'lightprobes/LP1.exr'" />
        <position x="..." y="..." z="..." />
        <cameraposition  x="..." y="..." z="..." />
        <cameraproperties   CFOVH="..." CFOVV="..." CRESH="..."
                            CRESV="..." ARES="..." f="..."
                            UPx="..." UPy="..." UPz="..." />
   </Lightprobe>
    ....
    <Insert as many lightprobes as necessary>
    ....
</Lightprobes>

<InputImages>
    <InputImage>
        <InputImage DEF="Input1_name" url="'inputimages/Input1.exr'">
        <lightprobe USE="LP1_name"/>
        <CameraVRML
                vrml_fovx="..."
                vrml_rotmag="..."
```

```
                vrml_rotvec="... ... ..."
                pos="... ... ..."/>
        </InputImage>
        ....
        <insert as many input images as necessary>
        ....
    </InputImages>
 </Scene>
</X3D>
```

The tag <Transform> </Transform> defines one material cluster, the tag <Shape> </Shape> defines an illumination cluster within a material cluster. The total number of transforms and the total number of shapes within a transform are unlimited. The tags Transform and Shape are used, as they are common in VRML nomenclature. The neutral geometry is also fed into a system as shapes and transforms, REFLECT distinguishes the difference between a material cluster and the neutral geometry through the missing element <InputImage USE=... /> within a <Shape> </Shape> environment. The attribute USE defines which input image should be used during the inverse illumination iteration to estimate the BRDF values. Therefore if a shape does not contain the <InputImage USE=... /> element, no BRDF values are calculated, or that shape is neutral geometry.

The element <Lightprobe></Lightprobe> defines the properties of a lightprobe image, such as its position. The element <InputImage></InputImage> defines the properties of an input image, such as its field of view. The number of input images is at most equal to the number of illumination clusters, the number of lightprobes is equal to the number of input images.

The elements of the X3D file are similar to that defined in the corresponding VRML file, but some elements and attributes were defined specifically for the system REFLECT. For clarity some of these are listed here:

- <SuggestedView.../> : the camera viewpoint of REFLECT after reading in the X3D file. This is defined by the position, view direction, up-vector, and field of view identities, defined within the SuggestedView tag.
- <texture url="..."/>: defines the location of the texture of the latitude-longitude image on the hard drive.
- <angulartexture url="..."/> [1]: defines the location of the texture of the lightprobe image on the hard drive.
- CFOVH, CFOVV: respectively the horizontal and vertical field of view of the camera used to capture the lightprobe image.
- CRESH, CRESV: respectively the horizontal and vertical resolution (number of pixels) of the camera used to capture the lightprobe image.
- ARES: the resolution that the sphere covers in the field of view of the camera, is equal to the dimension of the lightprobe image (as that image should be square).
- f: the focal length of the camera used to capture the lightprobe image.

[1] The name angular texture is actually not a good choice as it defines the location of the cropped lightprobe image, and not that of an angular map which, like the latitude-longitude image, is a specific image format.

- `UPx`, `UPy`, `UPz`: defines the up vector of the camera relative to the 3D geometric model of the scene, should be the same as the up vector in the 3D scene.
- `<position .../>`: the position of the lightprobe: if not defined, it is estimated by REFLECT.
- `<cameraposition .../>`: the position of the camera used to capture the lightprobe: if not defined, it is estimated by REFLECT.

The system REFLECT allows saving the scene with its intermediate results, such as the back-projected and diffuse textures. This saving operation adds some extra elements and attributes to the file, which are not listed in this appendix.

E.3 Scene control

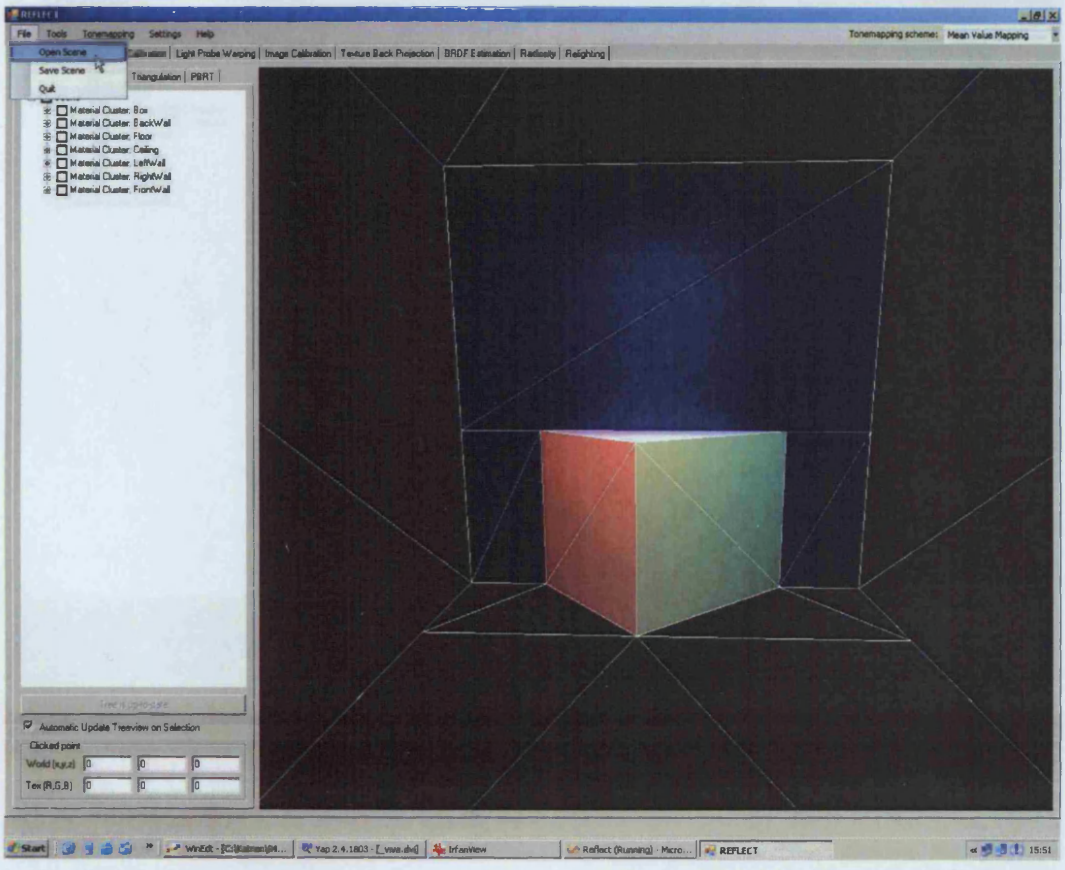


Figure E.1: *Clicking on FILE → OPEN opens an X3D file. The scene geometry and texture is rendered into the* ***viewing window*** *on the right hand side. The control tabs on the left allow analysing the scene. In the top right corner, the user can select various different types of tonemappers, to get the most optimal texture display.*

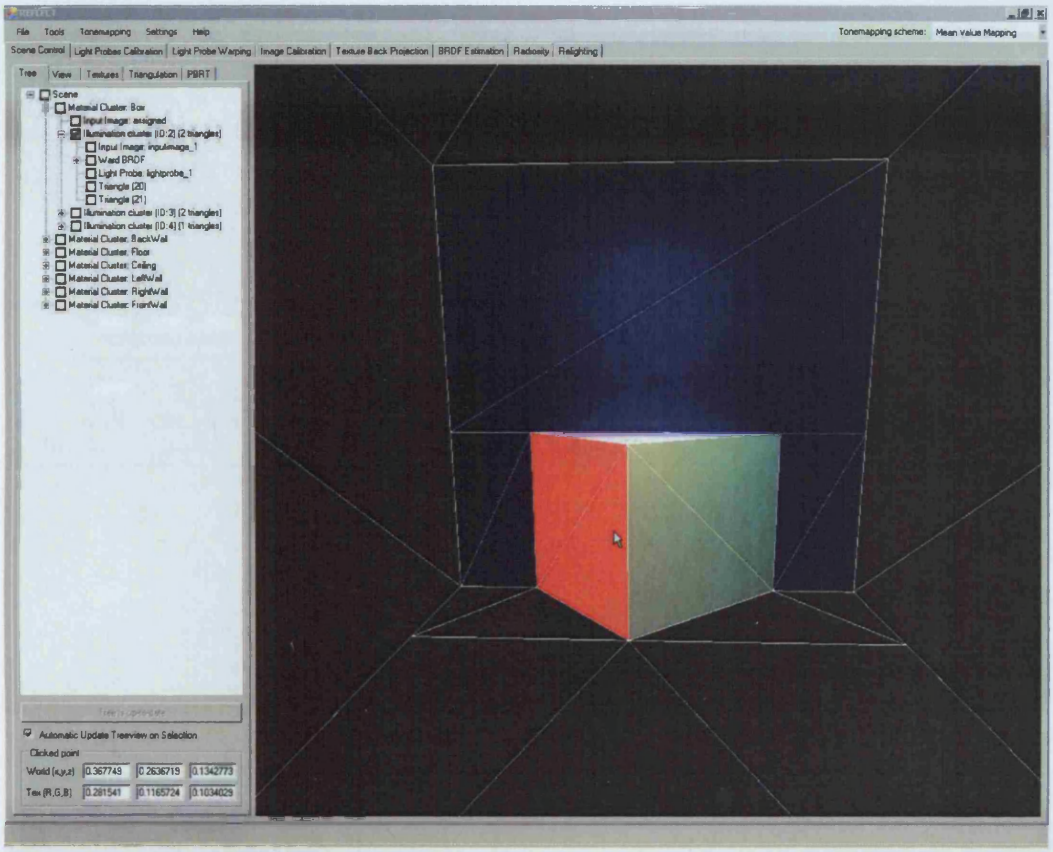


Figure E.2: *The* ***tree-view tab****, on the left hand side, presents the scene hierarchy into material and illumination clusters. Clicking on an IC in the scene in the* ***viewing window*** *on the right hand side, highlights that IC and unfolds useful information in the tree-view. Useful information is for instance, the current BRDF estimate, the lightprobe image used, the texture name, etcetera. In the example shown, the left side of the white object is highlighted.*

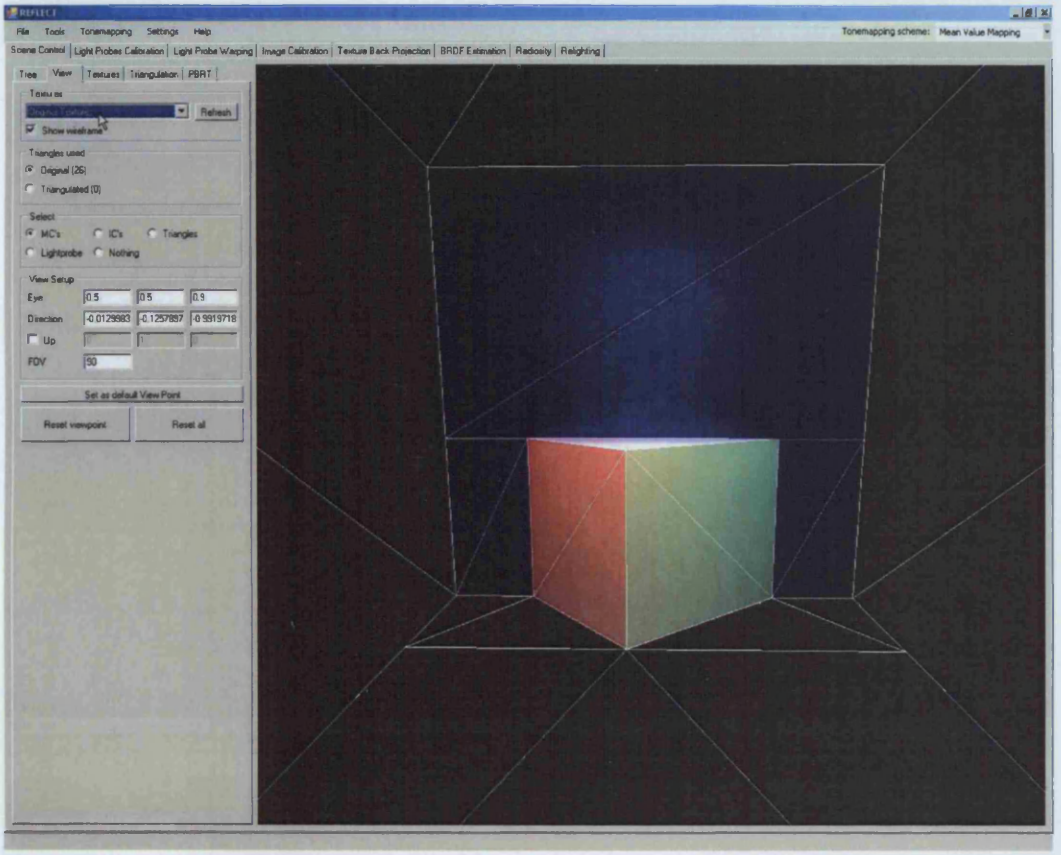


Figure E.3: *In the* ***view tab****, different textures can be selected to be projected onto the geometry. Different selection modes allow to analyse different aspects of the scene.*

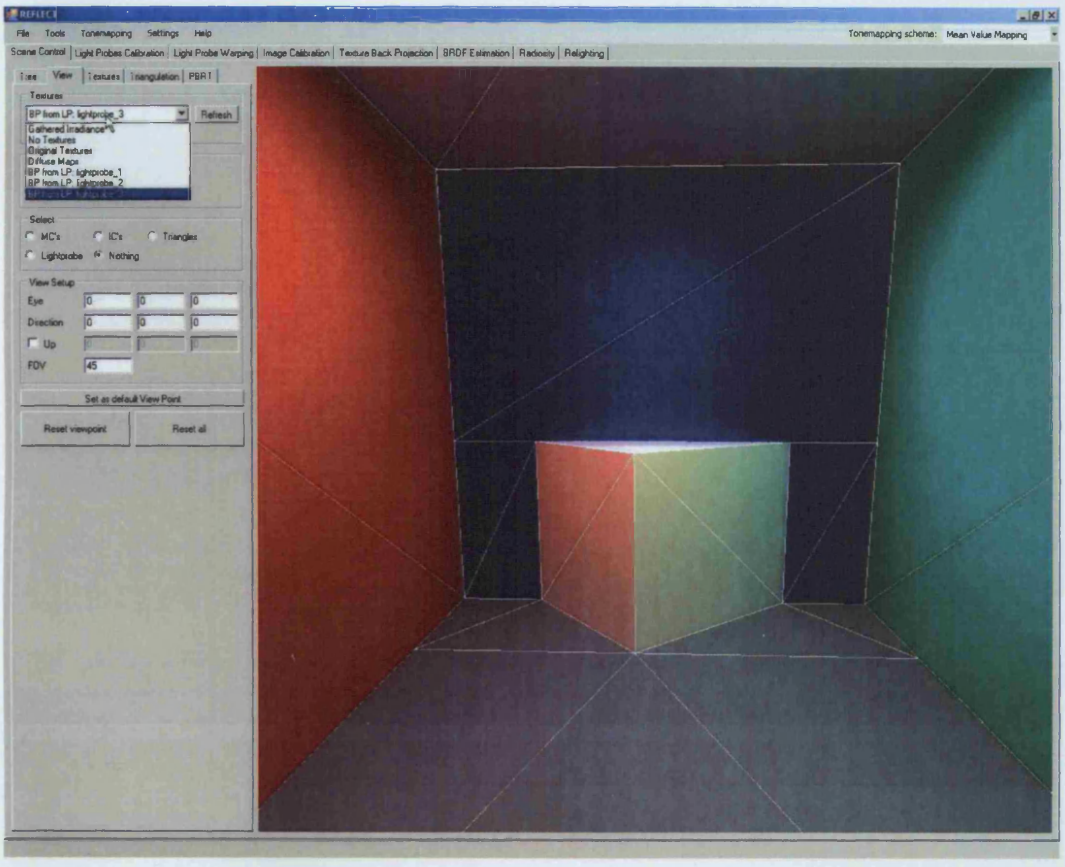


Figure E.4: *Instead of showing the original textures, the back-projected textures can be selected in the* ***view tab****. In the example shown, the back-projection of lightprobe 3 is shown. The user can choose between the original textures, the back-projected textures, the gathered irradiance (the amount of irradiance for each scene point calculated with a hemicube approach), and the diffuse textures to be shown.*

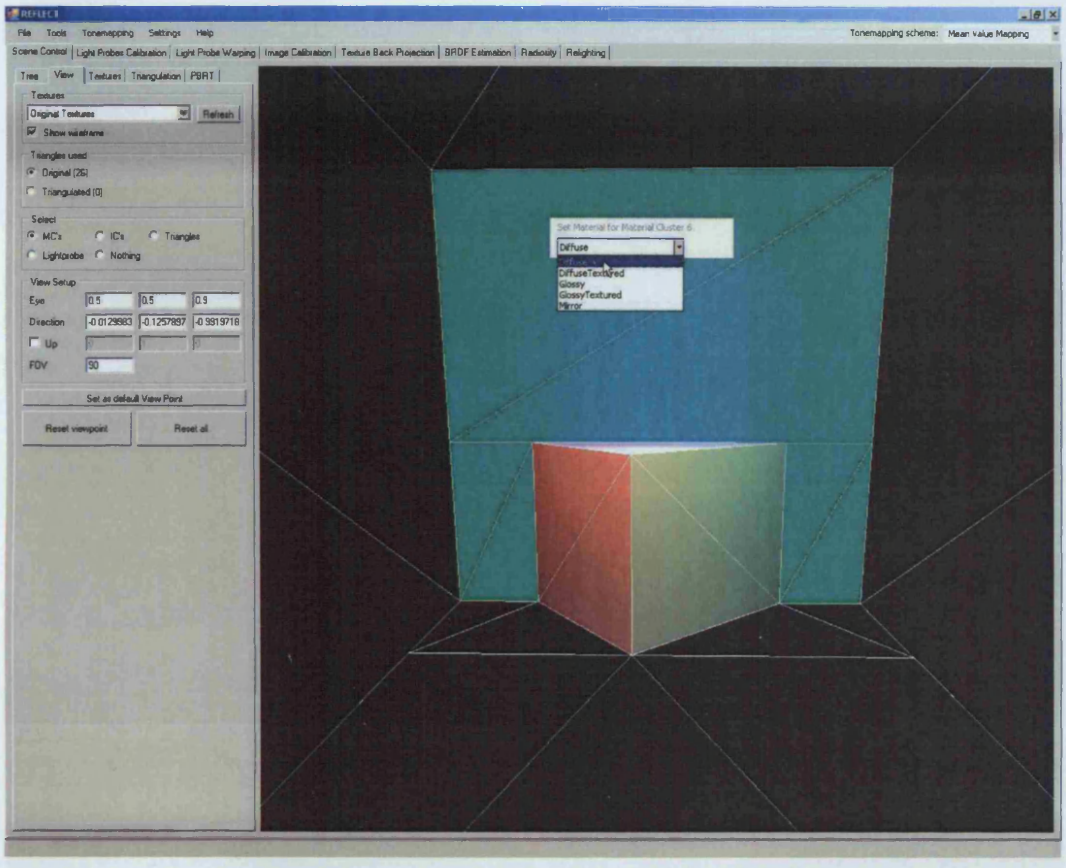


Figure E.5: *When the MC selection option is selected in the* ***view tab****, different MCs can be highlighted in the scene* ***viewing window****. Now, different types of BRDF can be assigned to the MCs (at this point only diffuse and diffuse textured can be selected). In this example, the blue MC is selected and highlighted in red. Its type of BRDF is set to diffuse. The default BRDF type is diffuse.*

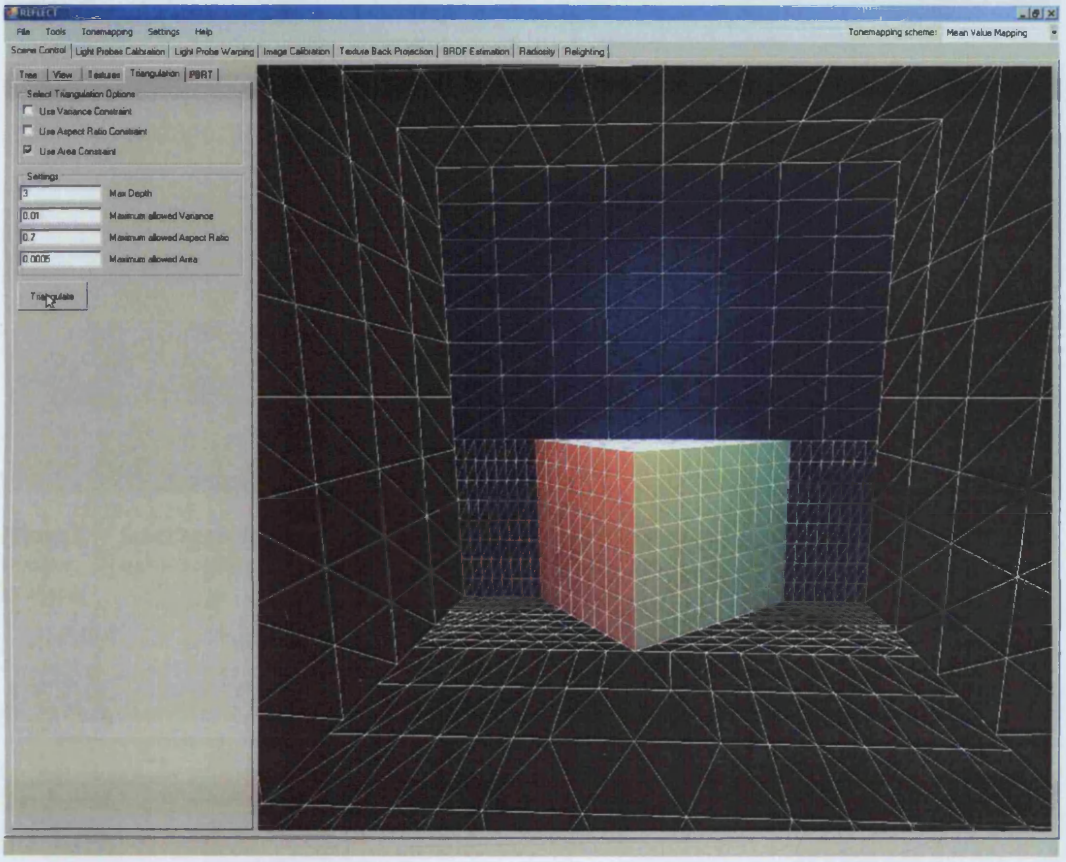


Figure E.6: *The* ***triangulation tab*** *lets the user manipulate the scene triangulation.*

E.4 Lightprobe calibration

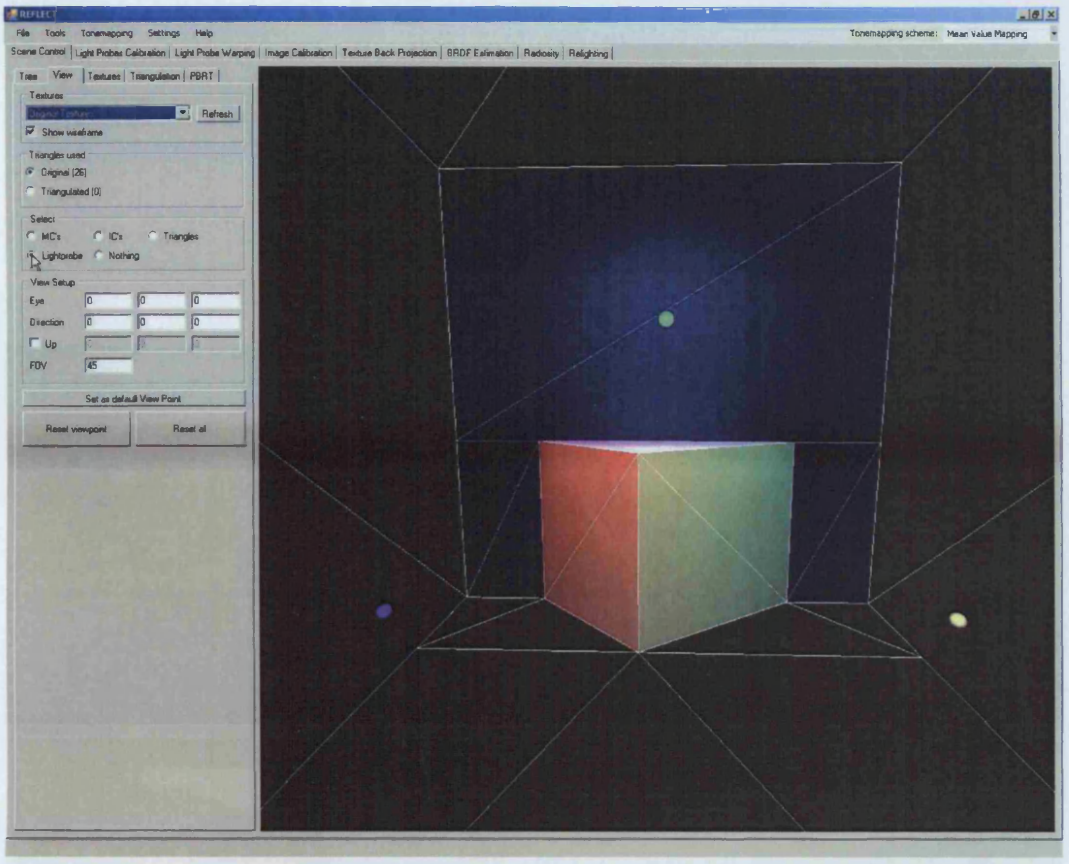


Figure E.7: *Selecting the* ***lightprobe*** *option in the* ***view tab*** *visualizes the position of the lightprobes in the* ***viewing window****. Though not illustrated here, if the user opens the* ***tree-view tab****, information about each lightprobe will be available.*

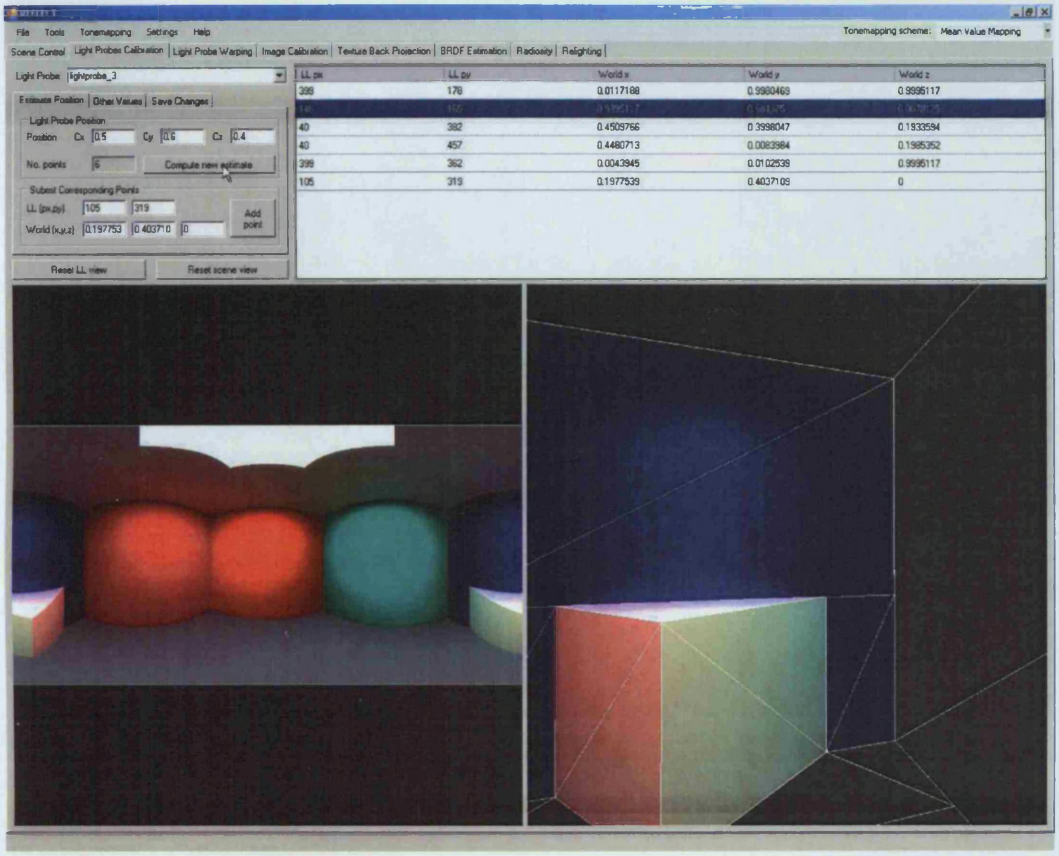


Figure E.8: *In the* ***lightprobe calibration tab*** *the user can select a lightprobe (lightprobe 3 in the example shown) and then select up to 30 different corresponding points between the lightprobe image and the 3D scene. After having selected 6 points or more, the user can calculate a lightprobe position.*

E.5 Lightprobe radiance registration

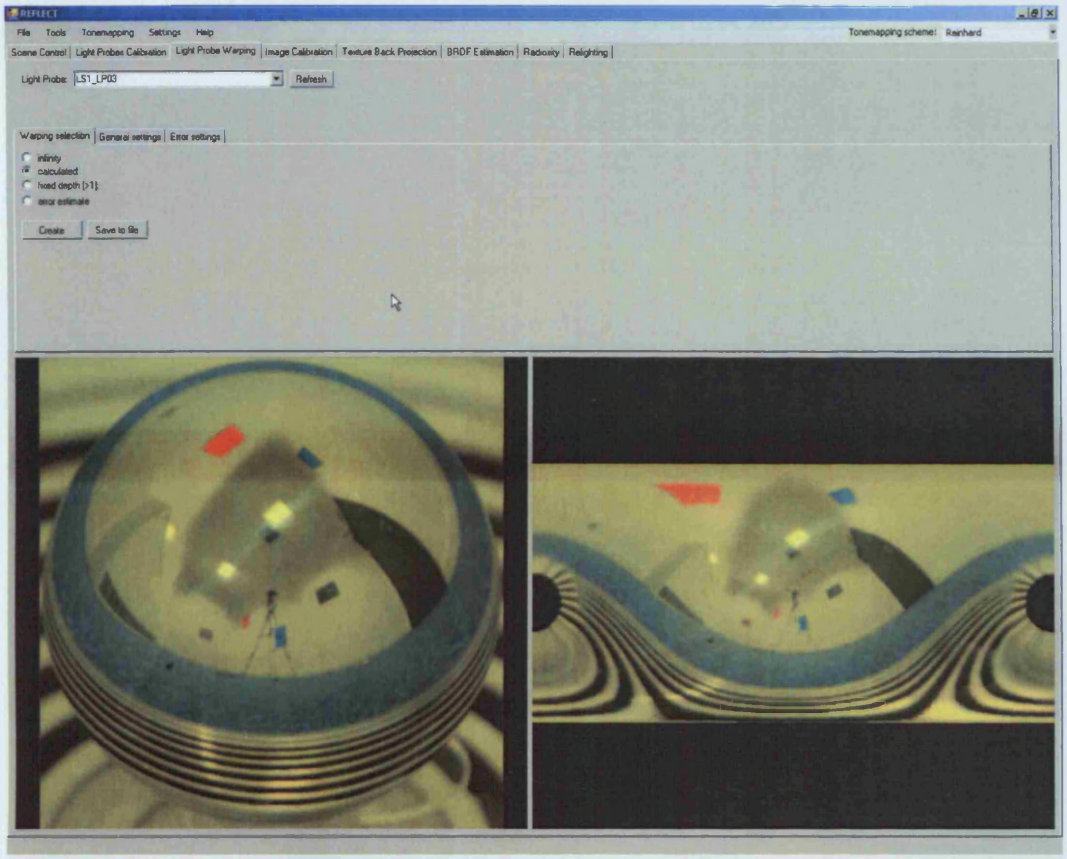


Figure E.9: *In the* ***lightprobe warping tab*** *a lightprobe image can be selected and warped to a correct latitude-longitude image. Other options, like error calculation, or applying an infinite scene or a scene at fixed depth, can be chosen as well.*

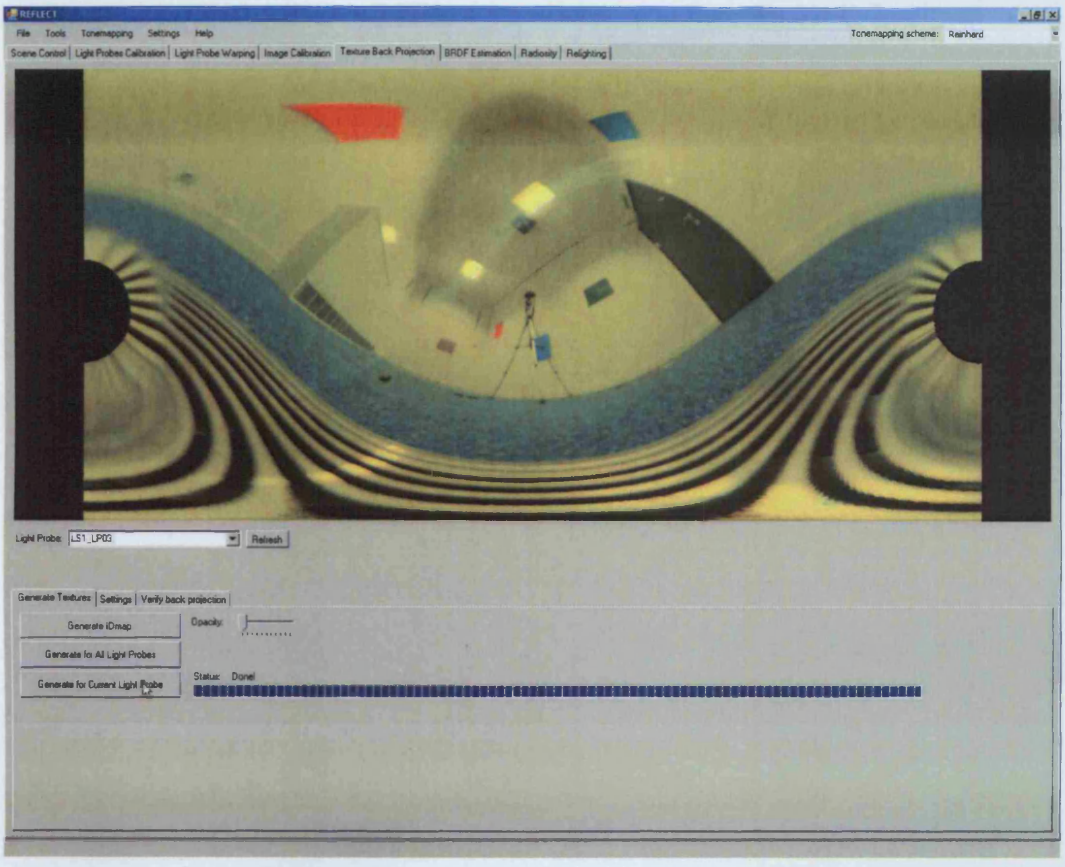


Figure E.10: *The* ***texture back-projection tab*** *back-projects the radiance values from a lightprobe image in latitude-longitude format onto the 3D geometry.*

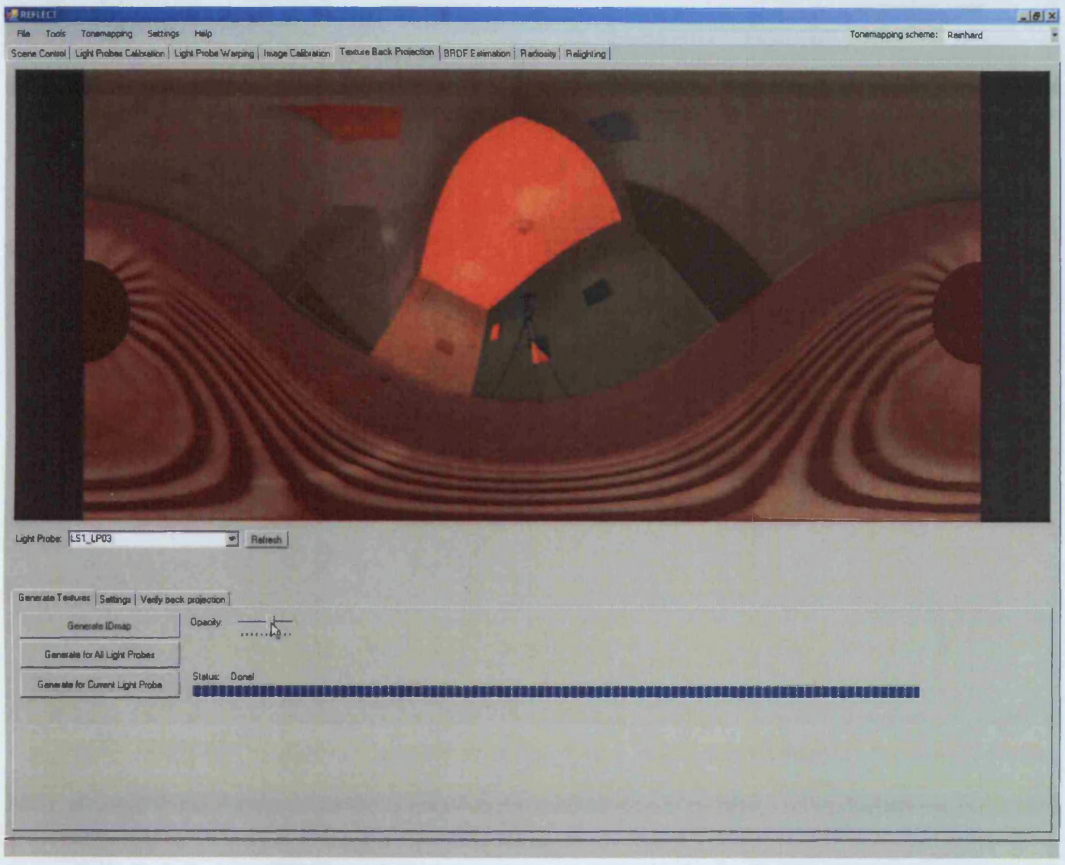


Figure E.11: *The* ***texture back-projection tab*** *allows the user to overlay the identity map over the latitude-longitude map. This helps the user decide if the lightprobe image and 3D scene overlap correctly.*

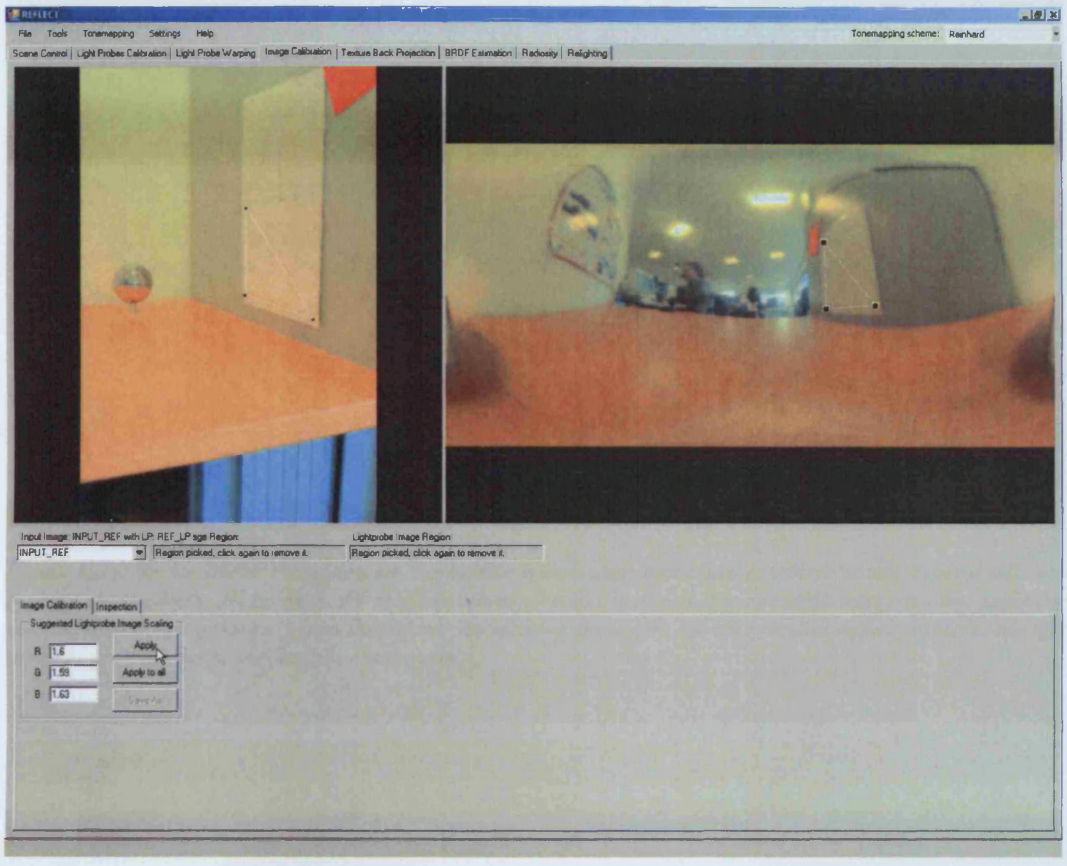


Figure E.12: *In the* ***image calibration tab*** *a reference input image and reference lightprobe image can be selected. The user clicks on three points in both images, which form two triangles. These triangles should roughly cover the same (usually white) object. Based on the radiance values in these triangles, a scaling factor is calculated. This scaling factor compensates for the fact that the reflective sphere is not* 100% *specular.*

E.6 Inverse Illumination

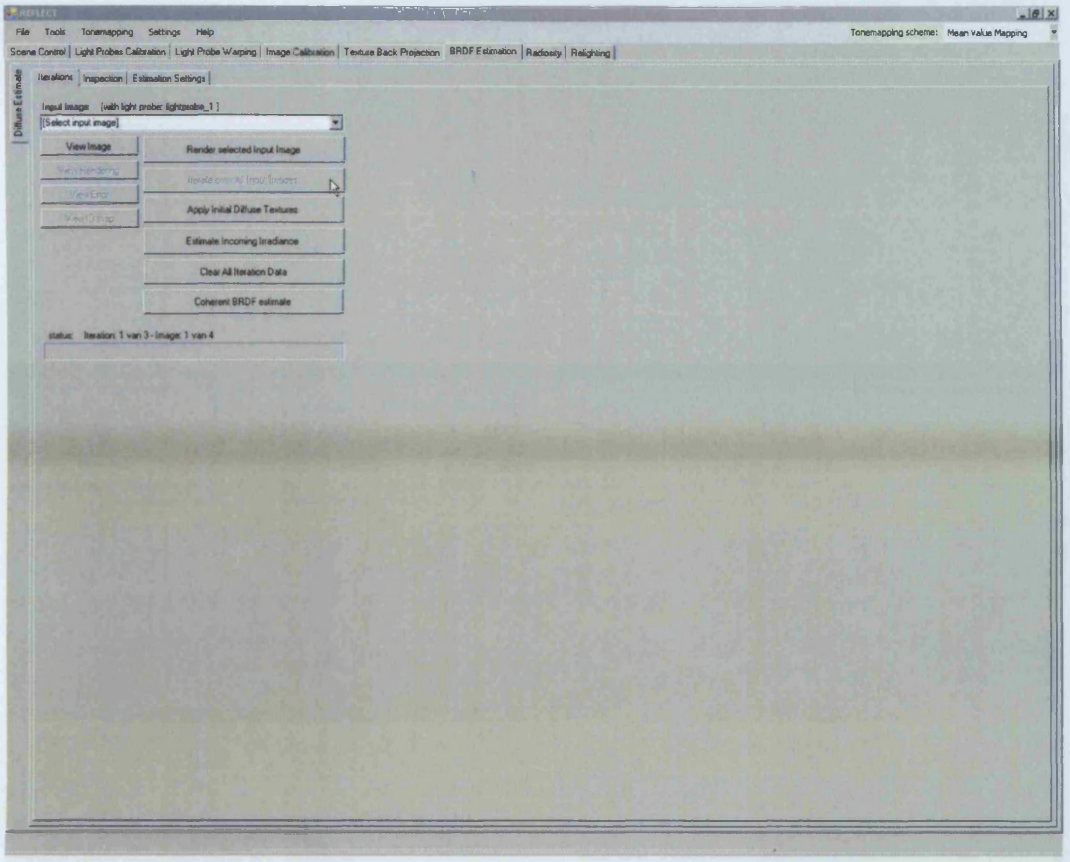


Figure E.13: *In the* ***BRDF estimation tab*** *the inverse illumination procedure can be initiated. Several different settings are available, for instance, the user can render a render image R_i for one input image I_i only, apply the coherency principle separately from the inverse illumination procedure, or apply initial diffuse textures. All this provides early feedback, and flexibility to the user.*

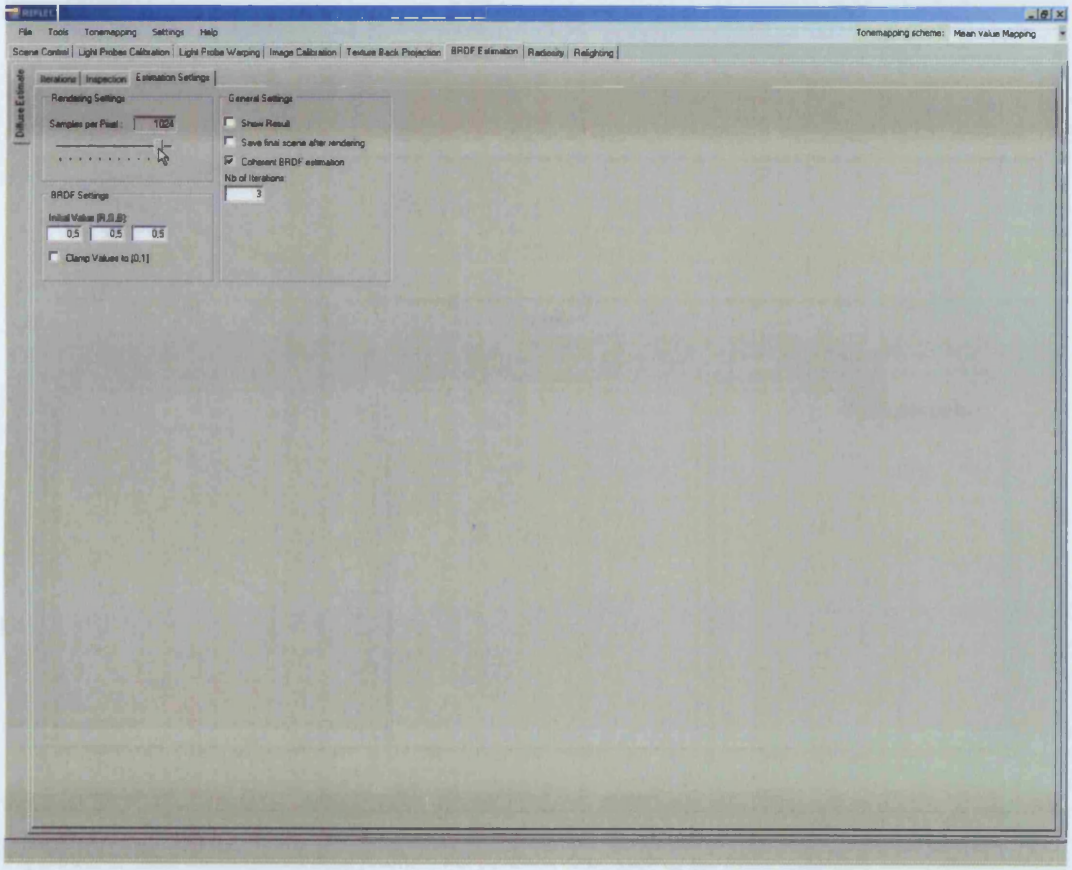


Figure E.14: *The* ***settings*** *tab inside the* ***BRDF estimation tab*** *allows a few more settings, such as the maximum number of iterations, the number of samples used per ray, and whether or not the coherency principle applies.*

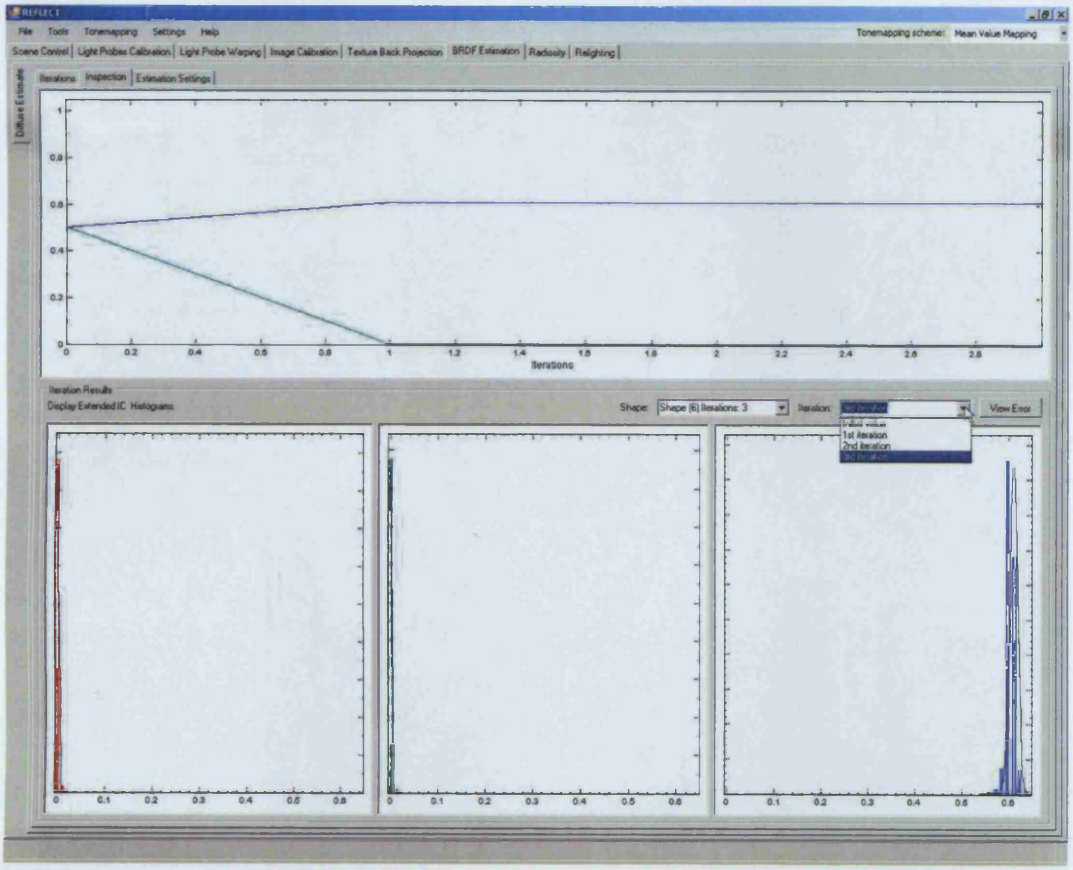


Figure E.15: *Once the iterations have finished, the iteration data, such as the BRDF estimates per illumination and material cluster and per iteration cycle can be investigated. If the coherency principle is not applied, a histogram of each illumination clusters gives an idea of the difference in BRDF estimate across a cluster. In the example shown, the BRDF values of a blue material cluster are investigated. The data shows that convergence was reached almost after the first iteration, but that the BRDF values varies over the MC, around 0.6 in the blue channel.*

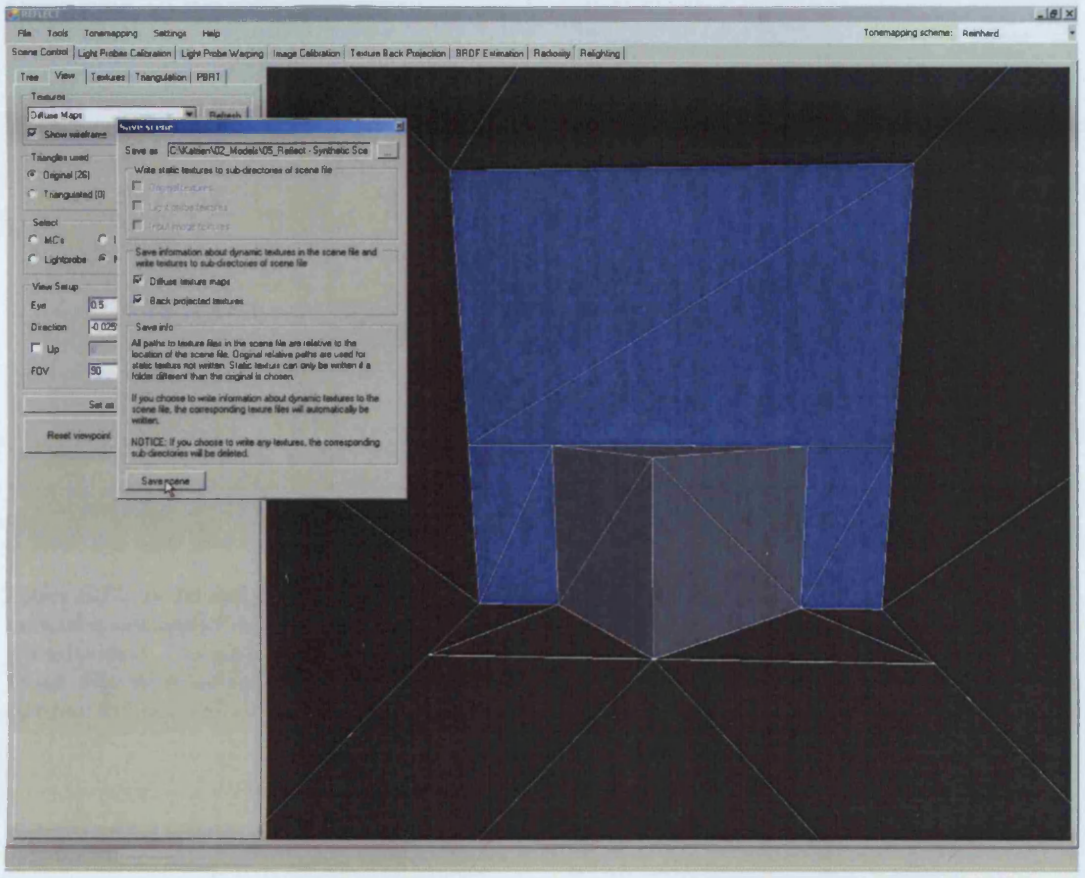


Figure E.16: *After having estimated the BRDF values, the user can save the scene data, including the back-projected textures, warped lightprobe images, and diffuse textures. This avoids having to go through these steps each time a relighting needs to be calculated. In the* ***viewing window*** *shown, the diffuse textures are shown, which allows an easy first examination of the BRDF values, prior to relighting.*

E.7 Relighting

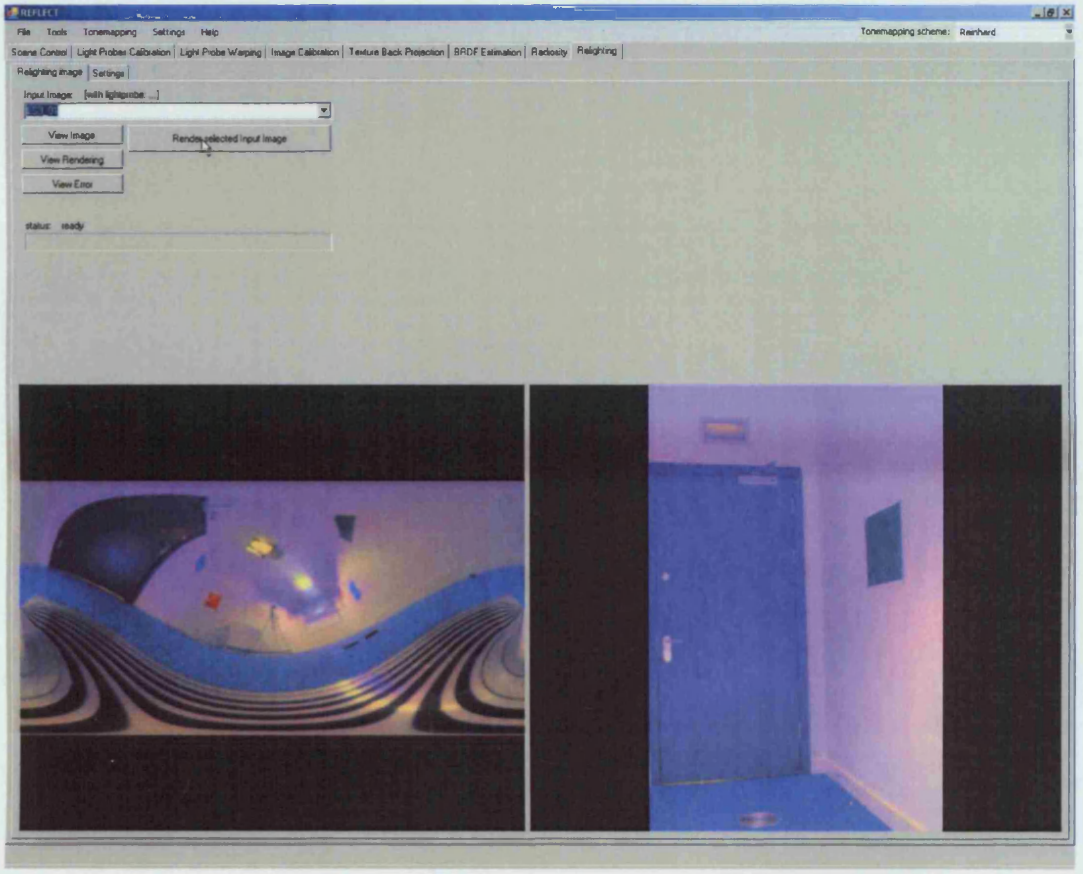


Figure E.17: *In the* ***Relighting tab*** *the user can select a lightprobe image that was not used during the BRDF estimation, and apply it to the scene for relighting. In the example shown, we see a novel lightprobe image and a reference image. The scene is now relit using the novel lightprobe image, from the same viewpoint as the reference image. After the relighting, the rendered image appears in the right window, to allow the comparison between the reference and rendered image.*